PLATAFORMA *WEB* DE INFORMAÇÃO AUTOMOBILÍSTICA

Cristiano Herlander Carvalho Alves

Departamento de Engenharia Electrotécnica

Instituto Superior de Engenharia do Porto

2013

Este relatório satisfaz, parcialmente, os requisitos que constam da Ficha de Disciplina de Tese/Dissertação, do 2º ano, do Mestrado em Engenharia Electrotécnica e de Computadores

Candidato: Cristiano Herlander Carvalho Alves, Nº 1080440, 1080440@isep.ipp.pt

Orientação científica: Cecília Maria do Rio Fernandes Moreira Reis, cmr@isep.ipp.pt

Departamento de Engenharia Electrotécnica

Instituto Superior de Engenharia do Porto

30 de Outubro de 2013

Agradecimentos

Quero desde já agradecer a preciosa colaboração da Professora Cecília Reis, pela sua disponibilidade, rigor científico e dedicação para que este projecto fosse um êxito.

Aos meus amigos e familiares agradeço todo o apoio que me permitiu abraçar este projecto, que sendo exigente, os torna ainda mais queridos. Quero deixar aqui um agradecimento especial à minha mãe pelos valores que sempre me tentou incutir e por me ter disponibilizado, na medida do possível, todo o apoio necessário à minha formação académica.

Estou ainda em dívida para com muitas pessoas pela sua ajuda, apoio e paciência. E é por isso que quero dedicar este trabalho a todos aqueles que, sem reservas, partilharam comigo os seus conhecimentos.

Resumo

Esta dissertação enquadra-se no âmbito dos Sistemas de Informação, em concreto, no desenvolvimento de aplicações *Web*, como é o caso de um *website*. Com a utilização em larga escala dos meios tecnológicos tem-se verificado um crescimento exponencial dos mesmos, o que se traduz na facilidade com que podem ser encontradas na Internet diversos tipos de plataformas informáticas. Além disso, hoje em dia, uma grande parte das organizações possui o seu próprio sítio na Internet, onde procede à divulgação dos seus serviços e/ou produtos.

Pretende-se com esta dissertação explorar estas novas tecnologias, nomeadamente, os diagramas UML - *Unified Modeling Language* e a concepção de bases de dados, e posteriormente desenvolver um *website*. Com o desenvolvimento deste *website* não se propõe a criação de uma nova tecnologia, mas o uso de diversas tecnologias em conjunto com recurso às ferramentas UML. Este encontra-se organizado em três fases principais: análise de requisitos, implementação e desenho das interfaces.

Na análise de requisitos efectuou-se o levantamento dos objectivos propostos para o sistema e das necessidades/requisitos necessários à sua implementação, auxiliado essencialmente pelo Diagrama de *Use Cases* do sistema. Na fase de implementação foram elaborados os arquivos e directórios que formam a arquitectura lógica de acordo com os modelos descritos no Diagrama de Classes e no Diagrama de Entidade-Relação. Os requisitos identificados foram analisados e usados na composição das interfaces e sistema de navegação. Por fim, na fase de desenho das interfaces foram aperfeiçoadas as interfaces desenvolvidas, com base no conceito artístico e criativo do autor. Este aperfeiçoamento vai de encontro ao gosto pessoal e tem como objectivo elaborar uma interface que possa também agradar ao maior número possível de utilizadores. Este pode ser observado na maneira como se encontram distribuídas as ligações (*links*) entre páginas, nos títulos, nos cabeçalhos, nas cores e animações e no seu *design* em geral.

Para o desenvolvimento do *website* foram utilizadas diferentes linguagens de programação, nomeadamente a *HyperText Markup Language* (HTML), a *Page Hypertext Preprocessor* (PHP) e *Javascript*. A HTML foi utilizada para a disposição de todo o conteúdo visível das

páginas e para definição do *layout* das mesmas e a PHP para executar pequenos *scripts* que permitem interagir com as diferentes funcionalidades do *site*. A linguagem *Javascript* foi usada para definir o *design* das páginas e incluir alguns efeitos visuais nas mesmas.

Para a construção das páginas que compõem o *website* foi utilizado o *software* *Macromedia Dreamweaver*, o que simplificou a sua implementação pela facilidade com que estas podem ser construídas. Para interacção com o sistema de gestão da base de dados, o MySQL, foi utilizada a aplicação *phpMyAdmin*, que simplifica o acesso à base de dados, permitindo definir, manipular e consultar os seus dados.

Palavras-Chave

Sistemas de Informação, Internet, Interfaces *Web*, Diagramas UML, Bases de Dados, Modelo Relacional.

Abstract

This dissertation fits within the scope of Information Systems, namely, in the development of *Web* applications, such as the case of a website. With the wide use of technological means it has been verified its exponential growth, which is reflected in the easiness in which they can be found on the Internet in every type of computer platforms. Moreover, nowadays, a large part of the organizations has its own website, to disclose your services and/or products.

The aim of this dissertation is to explore these new technologies, including Unified Modeling Language diagrams (UML) and design of databases, and then develop a website. The development of this website do not propose the creation of a new technology, but the use of different technologies together, using the UML tools. This process is organized in three main phases: requirements analysis, implementation and interface design.

In the requirements analysis was carried out the lifting of the proposed objectives for the system and the needs/requirements for its implementation, assisted essentially by the Use Case Diagram of the system. In the implementation phase were prepared the archives and directories that make up the logical architecture according to the models described in the Class Diagram and Entity-Relationship Diagram. The requirements identified were analysed and used in the interfaces composition and navigation system. Finally, in the interface design phase it were enhanced the interfaces developed, based on the artistic and creative concept. This enhancement fits the personal taste of its creator that aims to develop an interface which also appeals to as many users as possible. This can be seen in the way how are links distributed between pages, titles, headings, colors and animations and its overall design.

For the development of the website were used different programming languages, including HyperText Markup Language (HTML), Page Hypertext Preprocessor (PHP) and JavaScript. HTML was used for the disposal of all visible content of the pages and to define its layout and PHP was used to run small scripts that allow to interact with the different functions of the site. The JavaScript language was used to define the pages design and add some visual.

For the website pages construction was used the Macromedia Dreamweaver software, which simplified the implementation due to the facility in which can be constructed. For the interaction with the management system database (MySQL) it was utilized the phpMyAdmin application, which simplifies access to the database, allowing to define, manipulate and query their data.

Keywords

Information Systems, Internet, Web Interface, UML diagrams, Databases, Relational Model.

Índice

Índice de Figuras

Índice de Tabelas

Acrónimos

ADO – ActiveX Data Objects

AJAX – Asynchronous Javascript and XML

ASP – Active Server Pages

CSS – Cascading Style Sheets

DER – Diagrama Entidade-Relação

FTP – File Transfer Protocol

GPL – Gás de Petróleo Liquefeito

HTML – HyperText Markup Language

HTTP – HyperText Transfer Protocol

IEEE – Instituto de Engenheiros Eléctricos e Electrónicos

IMAP – Internet Message Access Protocol

JSP – Java Server Pages

MD5 – Message-Digest algorithm 5

NNTP – Network News Transfer Protocol

PHP – Page Hypertext Preprocessor

POP3 – Post Office Protocol 3

SI – Sistema Internacional de Unidades

SMTP – Simple Mail Transfer Protocol

SNMP – Simple Network Management Protocol

SQL – Structured Query Language

SSI – Server Side Include

UCD – User Centered Design

UML – Unified Modeling Language

URL – Uniform Resource Locator

WWW – World Wide Web

1. INTRODUÇÃO

Com a massificação da utilização da Internet, verificada a partir do início do século XX, os sistemas de informação das organizações passaram a estar disponíveis, cada vez mais, em multiplataforma (computador pessoal, *tablet*, *smartphone*, etc.) e nos diversos sistemas operativos existentes, como o *Windows* e o *Linux*. O meio utilizado para aceder aos sistemas das organizações é a própria Internet e os serviços são disponibilizados sob a forma de páginas *Web* acessíveis de qualquer parte do mundo.

Durante este processo evolutivo, o número de utilizadores e de *websites* cresceram de forma exponencial, tornando a *Web* acessível a todas as pessoas e com uma grande variedade de aplicações. Não obstante, muitos dos *sites* presentes na *Web* são visitados uma única vez pelos utilizadores, o que se verifica não porque o conteúdo do mesmo não lhes interessa mas sim porque estes não foram capazes de encontrar a informação desejada. Assim, facilmente se percebe a necessidade de um bom planeamento e da utilização de ferramentas de auxílio ao processo de criação de um *website*.

Uma das ferramentas mais utilizada para auxiliar este processo é a *Unified Modeling Language* (UML). A UML é uma linguagem gráfica padrão usada para especificação, construção, visualização e documentação de sistemas computacionais (secção 2.3 do capítulo 2). O processo de desenvolvimento de um *website* com recurso à UML é composto, essencialmente, por três fases: levantamento de requisitos, implementação e *design* das interfaces.

A fase de levantamento de requisitos compreende a definição dos objectivos, a procura das informações disponíveis para que haja um planeamento do *website* e o levantamento das necessidades do projecto e do sistema a implementar.

Na segunda fase, que representa a implementação, são seleccionados os requisitos definidos na etapa anterior para composição das interfaces e do sistema de navegação. Nesta efectua-se uma previsão dos níveis do localizador padrão de recursos (em inglês *Uniform Resource Locator* - URL) e dos locais de entrada do utilizador no *site*; elabora-se a documentação com informações pertinentes, as instruções de administração para os *web designers* e realizam-se testes às interfaces desenvolvidas a partir do servidor, de forma a verificar o correcto funcionamento das mesmas.

A terceira fase, que trata do *design* das interfaces, procura aprimorar o que foi desenvolvido nas fases anteriores, através do conceito artístico e criativo do(s) desenvolvedor(es) do sistema. Nesta fase o que predomina é a arte e a estética que podem ser observadas na distribuição das ligações (*links*) nas páginas, nas cores e animações, nos títulos, nos cabeçalhos e no *design* em geral.

Para o desenvolvimento de *websites* são utilizadas linguagens e tecnologias, modelos/metodologias e ferramentas gráficas e de *design*. Se um *website* for composto apenas por interfaces estáticas (conteúdo permanente) as necessidades do projecto centram-se na selecção das linguagens de programação (HTML, CSS, Javascript, etc.) e nas ferramentas de *design* adequadas. Se o *website* possuir interfaces dinâmicas (que sofram alterações constantes e que possam interagir com o utilizador) para além destas escolhas é necessário seleccionar a tecnologia responsável pela geração das páginas dinâmicas e do sistema de gestão da base de dados (secção 3.6). As linguagens para criação de páginas dinâmicas como a *Active Server Pages* (ASP), *Java Server Pages* (JSP) e *Page Hypertext Preprocessor* (PHP) são amplamente utilizadas.

Cada uma destas tecnologias é discutida com o detalhe apropriado nas secções 2.4 e 2.5, do capítulo 2, sendo apresentada uma breve explicação de cada tecnologia e citados exemplos para informação complementar. Apesar do esforço empregue, reconhece-se que algumas destas tecnologias mereciam uma maior profundidade de tratamento, no entanto acredita-se que esse esforço iria muito além dos objectivos propostos para este projecto.

O presente trabalho apresenta-se em duas divisões: a primeira relata sobre procedimentos, planeamento e estruturação de um *website*; a segunda parte apresenta um modelo funcional de um *website* com recursos às ferramentas UML.

1.1. OBJECTIVOS

O objectivo principal deste trabalho de dissertação é o desenvolvimento de uma plataforma *Web* (*website*) com recurso ao modelo UML, bases de dados e diferentes linguagens de programação (HTML, PHP, JavaScript, CSS, ASP, JSP). Para a concretização deste objectivo foi necessário desenvolver as tarefas a seguir apresentadas:

- estudo das Tecnologias de Informação (TI) focadas no desenvolvimento de aplicações Web;
- estudo dos diagramas *Unified Modeling Language* (UML) e do modelo relacional de bases de dados;
- modelação da plataforma *Web* :
    - desenvolvimento dos diagramas UML (diagrama de classes e diagrama de *use cases*) e do diagrama entidade-relação do modelo relacional de bases de dados;
    - desenvolvimento da base de dados relativa à plataforma *Web* criada;
- implementação da plataforma *Web*:
    - estudo das linguagens de programação usadas para desenvolvimento de aplicações *Web*;
    - estudo do *software Macromedia Dreamweaver* e da aplicação *phpMyAdmin*;
    - planeamento e selecção das ferramentas necessárias para o desenvolvimento do *website;*
    - desenvolvimento das diferentes páginas *Web* que compõem o *website*.

1.2. CALENDARIZAÇÃO

Sendo o desenvolvimento de uma plataforma *Web* a principal motivação deste trabalho, a sua prossecução conduziu à calendarização apresentada na Tabela 1. Esta inclui um conjunto de tarefas, como por exemplo: estudo das Tecnologias de Informação, estudo dos diagramas UML e do modelo relacional de bases de dados, modelação e implementação da

plataforma *Web* e elaboração do relatório, ordenadas de forma a ser seguida uma lógica sequencial.

1.3. ORGANIZAÇÃO DO RELATÓRIO

Este relatório encontra-se dividido em cinco capítulos. Neste primeiro capítulo efectua-se uma primeira abordagem ao projecto e suas especificações.

No capítulo 2 encontra-se o estado da arte, que resulta de uma pesquisa sobre os conceitos básicos associados à Internet e à criação de plataformas *Web*. Para uma melhor compreensão do ambiente *Web*, é apresentado o modo como este funciona, em que consiste, diferentes tecnologias que podem ser utilizadas para construir interfaces neste ambiente. É ainda efectuado um estudo acerca do funcionamento dos Diagramas UML e do Modelo Relacional de Bases de Dados, de forma a proporcionar uma maior facilidade de interpretação dos conteúdos apresentados no capítulo seguinte (Plataforma *Web* - Modelação).

Nos dois capítulos seguintes, capítulo 3 e capítulo 4, apresentam-se as várias etapas percorridas desde o planeamento até à implementação da plataforma *Web*, que podem ser agrupadas em dois grandes grupos: modelação e implementação.

Na etapa de modelação, apresentada no capítulo 3, é introduzido o conceito do projecto e efectuada a análise dos requisitos para a plataforma a ser desenvolvida. A seguir são apresentados os diagramas UML (diagramas de classes e de *use cases*) relativos à plataforma e o modelo relacional da base de dados, através da análise do diagrama entidade-relação. São ainda relatados todos os passos para o desenvolvimento da base de dados usada nesta plataforma, nomeadamente, a criação das diferentes tabelas de dados e interacção entre elas. O objectivo principal deste capítulo é expressar, numa linguagem modelo, as principais funcionalidades e relações afectas à plataforma.

No capítulo 4, referente à etapa de implementação, incluem-se as diferentes páginas que fazem parte do sítio e as potencialidades de cada uma delas, evidenciando todas as opções tomadas, como por exemplo as tecnologias usadas, e o modo como estas se relacionam entre si. De forma a possibilitar uma melhor compreensão do funcionamento da plataforma *Web* foram incluídos alguns códigos (extractos) utilizados para a programação destas páginas e as respectivas explicações do seu funcionamento.

Por último, no capítulo 5 são apresentadas as conclusões do trabalho desenvolvido e possíveis melhorias futuras.

Tabela 1 Calendarização do projecto

Etapa	Início	Fim	Duração	Agosto				Setembro				Outubro				Novembro			
				1ª	2ª	3ª	4ª	1ª	2ª	3ª	4ª	1ª	2ª	3ª	4ª	1ª	2ª	3ª	4ª
Estudo das Tecnologias de Informação	05-08-2013	07-08-2013	3 dias																
Estudo dos diagramas UML e do modelo relacional de base de dados	08-08-2013	11-08-2013	4 dias																
Modelação da plataforma *Web*	12-08-2013	25-08-2013	2 semanas																
Implementação da plataforma *Web*	26-08-2013	22-09-2013	4 semanas																
Elaboração do relatório	23-09-2013	25-10-2013	5 semanas																
Entrega do relatório	28-10-2013	01-11-2013	5 dias																
Apresentação do trabalho	----	11-11-2013	----																

2. ESTADO DA ARTE

As tecnologias de informação surgiram de uma forma laboratorial e em alguns casos até confidencialmente, sendo na actualidade amplamente difundidas, fazendo parte do quotidiano de cada um. De facto, estas tecnologias estão presentes em cada “recanto” para cumprir uma qualquer missão, e a sua utilização tornou-se rotineira e quase indetectável [28].

Uma das tecnologias de informação que mais evoluiu e que se assume como uma das mais utilizadas é a Internet. Esta distribui, através dos seus servidores, uma grande variedade de documentos (texto e multimédia) que qualquer utilizador da rede pode aceder e que apresentam, regra geral, uma ligação com outros serviços da Internet. Estes documentos formam a arquitectura *World Wide Web* (WWW), ou simplesmente *Web*, que tem facilitado a utilização em larga escala da Internet por todo o mundo, uma vez que qualquer utilizador, que possua um mínimo de conhecimento de informática, pode aceder à rede.

A *Web* foi criada na década de 90 pela Organização Europeia para a Pesquisa Nuclear, conhecida como CERN (antigo acrónimo para *Conseil Européen pour la Recherche Nucléaire*), como solução para os problemas de intercâmbio de informação entre os seus pesquisadores [8]. A partir daí, a tecnologia para construção de interfaces *Web* foi progressivamente incrementada permitindo o desenvolvimento de aplicações cada vez mais complexas. Entre estas interfaces *Web* destaca-se o desenvolvimento de *websites* (também conhecidos por *site* ou em português por sítio), que engloba a engenharia de

websites, bases de dados, metodologias e modelos, técnicas de uso de ferramentas gráficas e ainda outros conteúdos, conforme necessidades específicas [34].

Para o desenvolvimento de *websites* são utilizadas linguagens e tecnologias, modelos/metodologias e técnicas de uso de ferramentas gráficas e de *design*. Se o *website* for composto integralmente por interfaces estáticas (de conteúdo permanente) as necessidades do projecto centram-se na selecção das linguagens para programação e nas ferramentas de *design* adequadas. Caso possua interfaces dinâmicas (aquelas em que o conteúdo é gerado após uma consulta na base de dados, e apenas mantido durante a permanência do utilizador na interface), para além da escolha das linguagens e das ferramentas de *design* é ainda necessário seleccionar a tecnologia responsável por gerar as páginas dinâmicas e o sistema usado para a base de dados [16] [37] [43].

Para melhor entender os temas relacionados com o desenvolvimento de um *website*, que serão apresentados nas subsecções seguintes, é necessário entender como funciona a *Web*. Desta forma, será apresentado na subsecção 2.1.1 a arquitectura cliente-servidor, na qual a *Web* se baseia.

2.1. APLICAÇÕES *WEB*

Uma aplicação *Web* pode ser definida como um sistema informático que utiliza a *Web* como ambiente de execução, através de um navegador [34]. As aplicações *Web* envolvem frequentemente o desenvolvimento de *websites*, cuja arquitectura encontra-se representada na Figura 1. Nela é possível observar que quando o servidor *Web* recebe uma solicitação ("*pedido*") proveniente de um navegador, localiza o recurso num sistema de arquivos local e envia-o de volta para o cliente ("*resposta*"). Estes recursos podem ser documentos de texto, imagens, vídeos e áudio.

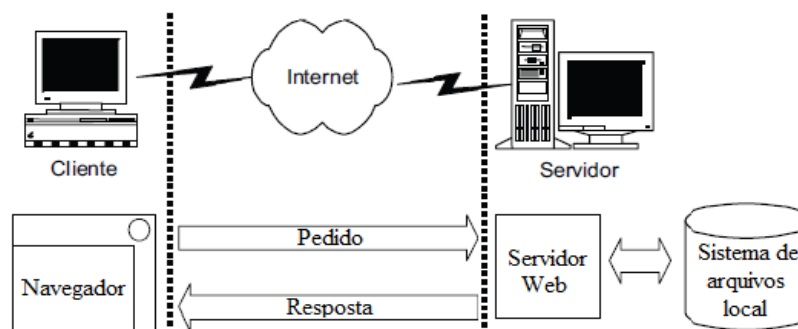



Figura 1 Arquitectura básica de uma aplicação *Web* [34]

Uma aplicação *Web* amplia o conceito de *website*, ao adicionar funcionalidade ao sistema, isto é, ao permitir que os utilizadores executem lógica de negócio através de um navegador. Esta deve ser entendida como uma forma de utilização de *software* através do acesso a dados persistentes a partir do servidor *Web*, permitindo ainda construir páginas dinâmicas para manipulação de dados.

2.1.1. ARQUITECTURA CLIENTE-SERVIDOR

Desde que foi criada, a *Web* evoluiu continuamente, o que é explicado, em grande parte, pela sua arquitectura simples porém eficiente [34]. Esta arquitectura é conhecida por "modelo cliente-servidor", e é constituída por um cliente e um servidor instalados sobre uma rede de computadores heterogénea, como mostra a Figura 2.

A arquitectura cliente-servidor é formada por um *browser* (conhecido em português por navegador) a funcionar do lado do cliente, que serve de intermediário para a solicitação de informações ao servidor e para as apresentar ao utilizador. O servidor, por sua vez, atende os pedidos dos diferentes clientes que compõem a rede e/ou outros servidores indistintamente. Pese embora, o modelo referencie uma arquitectura cliente-servidor, talvez fosse mais adequado apelidá-lo de pedido/resposta (*request*/*respost*), visto ser este o tipo de comunicação entre as partes: o cliente (navegador) solicita um documento ao servidor que processa o pedido, envia o documento ao cliente e encerra a comunicação. Esta arquitectura possui três componentes principais: o sistema de endereçamento, o protocolo de comunicação e a linguagem HTML (*HyperText Markup Language*).

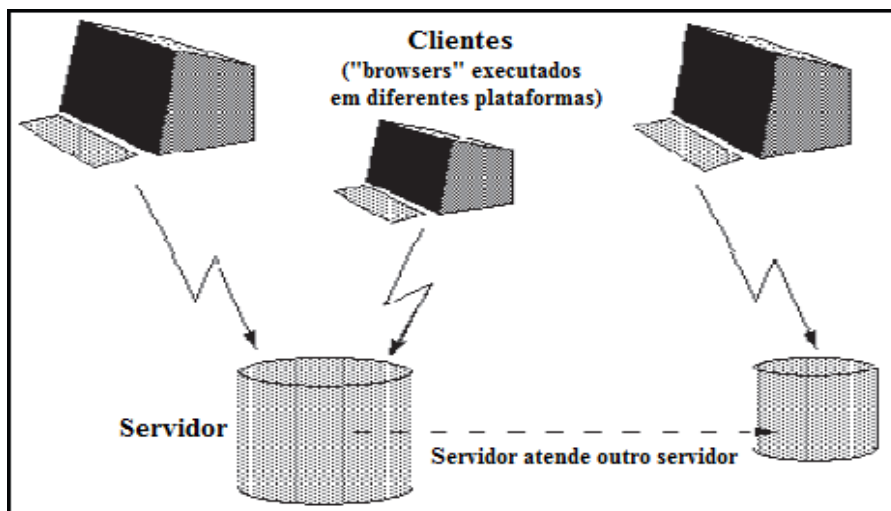



Figura 2 Modelo cliente-servidor da *Web* [34]

O sistema de endereçamento especifica o endereço (localização) do objecto/documento a que se pretende aceder e o protocolo usado para esse efeito. A URL (*Uniform Resource Locator*) é a forma mais conhecida para endereçamento de objectos via *Web* e apresenta a estrutura: *protocolo://servidor/recurso*, em que “*protocolo*” especifica o protocolo usado para a comunicação, “*servidor*” designa o servidor e “*recurso*” o documento ou recurso solicitado.

O protocolo mais utilizado para comunicações do tipo cliente-servidor é o HTTP (*HyperText Transfer Protocol*). Este especifica o modo como são realizadas as transacções de informação entre o(s) cliente(s) e o(s) servidor(es), através da utilização de algumas regras básicas. Outros protocolos com um número de utilizações significativo são o FTP (*File Transfer Protocol*), usado para envio de recursos de um computador para um servidor na *Web*, e o SMTP (*Simple Mail Transfer Protocol*) usado para correio electrónico.

A linguagem HTML define a estrutura e a formatação dos documentos do tipo hipertexto através de etiquetas (*tags*) que indicam o modo como estes devem ser visualizados. As principais vantagens da utilização desta linguagem são: a facilidade em associar as informações através de *links* (permitindo a formação de grandes redes de informação), o mecanismo de navegação simplificado e uniforme e a facilidade com que esta pode ser percebida, mesmo por utilizador não especializados.

2.2. DESENVOLVIMENTO DE *WEBSITE*

Nos primeiros *websites* desenvolvidos era dada pouca importância à análise dos seus requisitos, o que originava que as interfaces criadas não atendessem às necessidades dos seus utilizadores. Esta falha do processo de desenvolvimento foi mais tarde parcialmente colmata com a introdução de técnicas centradas no utilizador, denominadas por *User Centered Design* (abreviadamente UCD), o que permitiu a inclusão dos requisitos dos utilizadores durante o projecto do *site*.

Contudo esta mudança de atitude era ainda insuficiente para a obtenção de *websites* com qualidade, uma vez que o seu *design* era ainda elaborado de forma muito informal. Presentemente, os projectistas socorrem-se de ferramentas para edição de páginas e/ou para identificação dos requisitos, no entanto, estas não fornecem suporte a muitas fases do ciclo de concepção de um *website*. Embora a prática informal seja aceitável para projectos de

pequena dimensão (projectos do tipo "*do your own web page*"), esta torna-se critica para *sites* de maior complexidade e/ou dimensão.

Embora não haja um consenso do modo como deve ser sistematizado o processo de criação de um *website* e quais os modelos necessários para este processo, existe um padrão conhecido como "*IEEE-2001: Práticas recomendadas para a Internet – Engenharia de websites, gestão de websites e ciclos de vida de websites*", definido no IEEE (Instituto de Engenheiros Eléctricos e Electrónicos) [21], que inclui um conjunto de normas para o desenvolvimento de *websites* e os tipos de informação que nestes devem ser destacados.

Algumas destas normas que pretendem facilitar o sistema de acesso e navegação de um *website* são:

- todos os documentos e informação diversa publicados num *website* deverão, terminado o seu "prazo de vida", passar para o arquivo e manter-se disponível para os utilizadores, por exemplo, através de motores de pesquisa, de forma a facilitar a procura de informação;
- devem ser realizados testes sobre as páginas em diferentes navegadores e em diferentes versões, pelo menos para os dois navegadores com maior divulgação e nas suas duas últimas versões;
- A concepção das páginas deverá permitir a navegação mesmo com a resolução gráfica diminuta.

2.2.1. ETAPAS (PLANEAMENTO)

Antes de se criar um *website* este deve ser devidamente planeado, isto é, devem ser identificados os recursos técnicos, financeiros e humanos necessários para o acompanhamento, desenvolvimento e promoção do projecto. Sem um bom planeamento corre-se o risco de o *website* criado não corresponder às expectativas do seu público-alvo ou cliente final [37].

O ciclo de desenvolvimento de um *website* é uma espiral contínua, marcado por sucessivas modificações. Ao longo deste ciclo vão-se sucedendo várias etapas, sendo que o número e a importância de cada uma delas variam em função da abordagem utilizada. A seguir será apresentada e descrita a abordagem proposta por Scapin que é composta por seis etapas, como mostra a Figura 3. Estas incluem: análise de requisitos, especificação, *design*, implementação, testes/avaliação e manutenção [38].

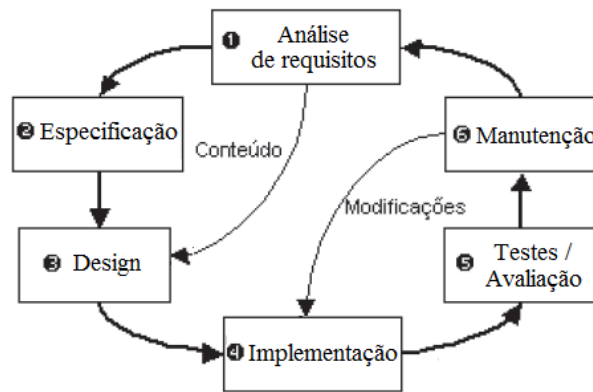



Figura 3 Ciclo de desenvolvimento de um *website* – modelo proposto por Scapin [38]

O modelo proposto por Scapin para o desenvolvimento de um *website* configura-se como um processo sistemático e cíclico, onde deve ser executada, em primeiro lugar, a análise dos requisitos. Esta etapa diz respeito à estruturação do *site* e identificação do seu contexto, definindo os objectivos do mesmo, caracterizando o seu público-alvo e identificando os recursos que serão utilizados. Na fase de especificação são produzidos os modelos da interface, a partir dos requisitos obtidos durante a análise de requisitos. A seguir estes modelos são refinados e é implementado o *website*, de acordo com o seu conteúdo (*Design*). A etapa da implementação corresponde à criação das páginas HTML e dos objectos de som/imagem necessárias para desenvolvimento da aplicação.

Embora os testes ocorram durante todo o processo de criação de um *website*, o teste definitivo acontece após concluída a implementação do mesmo, ou seja, na etapa de testes/avaliação. Nesta são testados *links*, imagens, textos, ligações com a base de dados, formulários e outros elementos interactivos, para além do teste da imagem do *website* em diferentes plataformas e *browsers*. Por fim, a fase de manutenção envolve a recolha de novos requisitos e o planeamento das modificações necessárias (identificadas na fase de testes).

De referir que este ciclo engloba ainda um “atalho” que possibilita a passagem à etapa da implementação logo após a análise de requisitos, sem se passar pela etapa de especificação, o que na prática se verifica frequentemente. Porém, isto dificulta o processo de criação do *website*, uma vez que são necessárias modificações frequentes do mesmo sempre que seja alterada a sua interface (devido a não ter sido criado um modelo da interface na etapa da especificação).

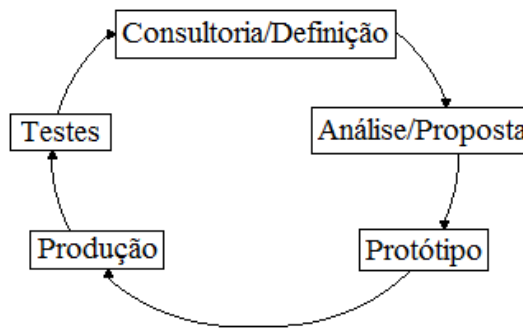



Figura 4 Ciclo de desenvolvimento de um *website* – modelo proposto por Santos

Para além do modelo proposto por Scapin existem muitos outros modelos de referência para o desenvolvimento de um *website*. Um modelo que merece ainda referência neste trabalho é o proposto por Santos [37], que sugere uma divisão deste processo em cinco etapas principais: consultoria/definição, análise/proposta, protótipo, produção e testes (Figura 4).

A primeira etapa (consultoria/definição) compreende um trabalho conjunto entre o desenvolvedor e o cliente (quando possível) no planeamento e organização do projecto. Após a fase de consultoria é elaborado um relatório detalhado com as exigências técnicas para o *website* e a estrutura básica de navegação (análise/proposta). A seguir à aprovação da proposta é apresentado o protótipo do *website*, que pode incluir apenas o *layout* do mesmo ou em alguns casos um projecto semi-funcional. Quando o *layout* se encontrar de acordo com as expectativas do cliente, é então possível a produção das imagens, textos, formulários e outros elementos interactivos (etapa de produção). Para tal, o trabalho é, regra geral, dividido internamente entre os envolvidos no processo. Por último, efectuam-se testes ao *website* desenvolvido e executam-se as alterações necessárias.

2.2.2. TIPOS DE PERFIS

No que diz respeito à elaboração de um *website* não existe um modelo único, devido ao facto de estes variarem de acordo com a natureza e o propósito do mesmo. Estes constituem-se, normalmente, pelos esforços de um único indivíduo ou de uma empresa/organização, e são dedicados a um tópico ou propósito em particular. É sempre complicado afirmar com clareza até “onde vai um *website*”, havendo alguns *websites* que são autênticas “mantas de retalhos”, isto é, complicados e de navegação algo confusa, o que origina que os seus visitantes se percam antes de conseguirem aceder à informação desejada [6].

Os perfis mais utilizados na construção de *websites* são:

- comunidades virtuais: servem para os utilizadores comunicarem com outros utilizadores e surgem, normalmente, associados a um tema criado e mantido pela própria dinâmica da interacção entre os seus visitantes. Nesta categoria incluem-se os fóruns, *chats,* sites de relacionamento, etc.;
- institucional: este perfil é bastante utilizado por empresas como ponto de contacto com os seus clientes e/ou fornecedores, a fim de aumentar a sua repercussão no mercado, gerando potencial de negócio e informações. Neste perfil incluem-se ainda os *websites* criados por profissionais liberais para publicação dos seus trabalhos;
- portal: congregam conteúdos de diversos tipos de perfis, geralmente fornecidos por uma empresa ou organização. O nome atribuído a este tipo de perfil prende-se com o facto de estes congregarem uma grande quantidade de serviços num único local;
- produtos: são utilizados por instituições comerciais para venda dos seus produtos, apresentado as suas características e benefícios;
- serviços: à semelhança do perfil anterior, este também é utilizado pelas instituições porém para venderem serviços de consultoria, formação, entre outros. O seu conteúdo visa essencialmente esclarecer o domínio dos serviços prestados por essa instituição e "dar a conhecer" os mesmos aos visitantes.

2.3. Diagramas UML e Modelo Relacional de Bases de Dados

Entre o final da década de 80 e inícios da década de 90 (sensivelmente entre os anos 1989 e 1994) a quantidade de métodos orientados a objectos existentes aumentou consideravelmente, o que ocasionou que os seus utilizadores tivessem dificuldades em encontrar uma linguagem de modelação capaz de atender às suas necessidades [16]. Tal tornou necessário o aparecimento de uma linguagem unificada em que os desenvolvedores de *softwares* pudessem "falar" numa linguagem comum [12].

Com o intuito de encontrar uma solução para os problemas decorrentes no desenvolvimento de *softwares,* os autores Grady Booch, Ivar Jacobson e James Rumbaugh uniram-se e unificaram os seus métodos, criando a *Unified Modeling Language* (UML) [9]. A UML é uma linguagem gráfica destinada à especificação, construção, visualização e

documentação de sistemas computacionais através do paradigma de Orientação a Objectos. Esta linguagem tornou-se, nos últimos anos, a linguagem padrão para modelação de *softwares*, tendo sido adoptada internacionalmente pela indústria de Engenharia de Software [20].

De referir que a UML não é uma linguagem de programação, mas sim uma linguagem de modelação, cujo objectivo é auxiliar os desenvolvedores de *softwares* a definir as características dos *softwares*, bem como os seus requisitos, estrutura lógica e até os equipamentos sobre os quais o sistema deverá ser implementado, antes de se iniciar o desenvolvimento do *software*.

A UML possui catorze tipos de diagramas, divididos em duas grandes categorias: estruturais e comportamentais. Sete destes diagramas representam informações estruturais e os outros sete informações gerais de comportamento. Na Figura 5 podem ser visualizados de forma hierárquica os diferentes diagramas UML existentes. O objectivo de ser disponibilizado um número relativamente elevado de diagramas é fornecer múltiplas "visões" do sistema a ser modelado, possibilitando a análise e modelação sob diferentes aspectos, procurando atingir uma modelação o mais completa possível (cada diagrama complementa os outros diminuindo a possibilidade da ocorrência de erros durante o desenvolvimento do *software*). Embora cada diagrama tenha a sua utilidade nem sempre é necessário modelar um sistema utilizando todos estes diagramas, uma vez que alguns deles possuem funções muito específicas.

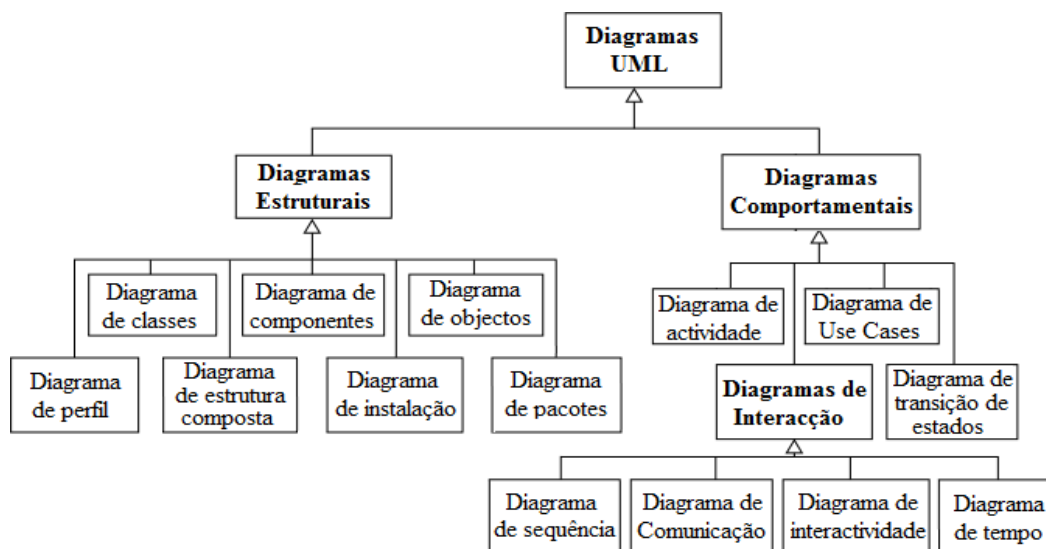



Figura 5 Hierarquia dos diagramas UML existentes

Nas subsecções seguintes, 2.3.1 e 2.3.2, serão descritos de uma forma breve, destacando os seus objectivos principais e as suas características, os diagramas UML com relevo para o projecto desenvolvido neste trabalho, isto é, diagrama de *use cases* e de classes, respectivamente. Na subsecção 2.3.3 é apresentado o diagrama entidade-relação, do modelo relacional de bases de dados.

2.3.1. DIAGRAMA DE *USE CASES*

O diagrama de *use cases* (em português, diagrama de casos de uso) corresponde ao diagrama mais geral e informal da UML, sendo utilizado como auxílio aos processos de levantamento e análise dos requisitos, e na compreensão do sistema como um todo [20]. Apesar do seu carácter geral e informal este serve de base para os restantes diagramas e é consultado durante todo o processo de modelação, visto conter uma descrição das principais funcionalidades do sistema (sob o ponto de vista dos utilizadores).

Um diagrama de *use cases* é composto por: actores, casos de uso e relações.

- Actores:

Um actor é uma entidade externa (fora do sistema) que interage com o sistema, e pode ser um utilizador humano, um outro sistema físico ou lógico ou um evento externo. A representação gráfica de um actor caracteriza-se por um boneco e por um rótulo com o seu nome, conforme apresentado na Figura 6. Os actores devem ser caracterizados por intermédio de uma pequena descrição, de modo a assegurar uma boa compreensão do seu significado por parte de todos os elementos da equipa envolvida na análise [2].

Os actores comunicam com o sistema através dos casos de uso, podendo invocar vários casos de uso (cada caso de uso pode ser invocado por vários actores).

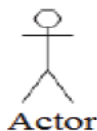



Figura 6 Diagrama de *use cases*: representação gráfica de um Actor [2]

- Casos de Uso:

Após serem identificados os actores que compõem o sistema, pode-se identificar os casos de uso (ou *use cases*) com os quais cada actor interage com o sistema. Um caso de uso permite caracterizar as funcionalidades que determinada aplicação a ser desenvolvida deve disponibilizar aos seus utilizadores.

Devido ao facto de nem todos os *use cases* virem a ser suportados por um sistema informático estes podem ser definidos segundo uma perspectiva de negócio, pela identificação do modo como respondem a um cliente ou evento em termos de processo de negócio.

Quando se implementa um caso de uso, a responsabilidade de cada passo da execução é associada às classes (actores ou casos de uso) que fazem parte da colaboração, através da especificação das operações necessárias e da definição do modo como estas irão interagir com o sistema. A representação gráfica de um caso de uso caracteriza-se por uma elipse e por um rótulo com o seu nome, como apresentado na Figura 7.

Figura 7 Diagrama de *use cases*: representação gráfica de um Caso de Uso [2]

- ❖ Relações:

Os *use cases* podem estar relacionados entre si. As relações mais comuns entre use cases são: generalização, "*include*" e "*extend*". A relação de generalização especifica que um *use case* (UC) herda as características de um outro UC, e pode sobrepor algumas delas ou adicionar novas características. Nesta situação, diz que o primeiro UC é um caso particular do segundo (conhecido por "Super" Caso de Uso). A relação "*include*" significa que para determinado UC ter a(s) sua(s) funcionalidade(s) executada(s) necessita de utilizar ou incluir uma funcionalidade de um outro UC. A relação "*extend*" significa que deve ser incluído num determinado UC um comportamento opcional, que se encontra definido num outro UC base.

Existem ainda relações entre actores e *use cases*, que são apelidadas de associação (Figura 8) e definem uma funcionalidade do sistema sob o ponto de vista do utilizador. Podem também existir relações entre actores, denominadas de generalização. À semelhança do que se verifica na relação de generalização entre UC, também nesta um actor herda as características de um outro actor (dos seus *use cases*). Esta relação encontra-se representada na Figura 8, onde os UC do actor B são também os UC do actor A, no entanto, o actor A possui os seus próprios UC.

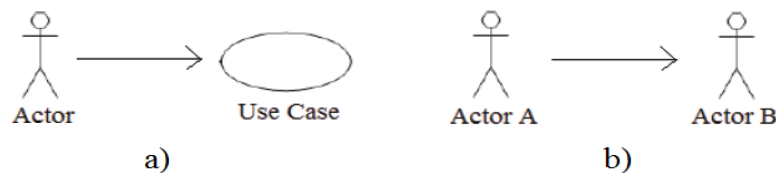



Figura 8 Diagrama de *use cases*: a) exemplo de uma relação de associação ; b) exemplo de uma relação de generalização [2]

2.3.2. DIAGRAMA DE CLASSES

Os diagramas de classes são os mais utilizados e por ventura os mais importantes no contexto dos diagramas UML, servindo de apoio à maioria dos outros diagramas. Como o próprio nome indica, estes diagramas definem a estrutura das classes utilizadas no sistema, determinando os atributos e métodos usados por cada uma delas, para além de estabelecer o modo como se relacionam e trocam informações entre si. A sua criação resulta de um processo de abstracção onde são identificados os objectos (entidades e conceitos) relevantes para o sistema a ser modelado e onde se procura descrever as características comuns em termos de propriedades (atributos) e de comportamentos (operações).

Os diagramas de classes são considerados estáticos, uma vez que a estrutura neles descrita é sempre válida em qualquer ponto do ciclo de vida do sistema. Frequentemente, os sistemas possuem mais do que um diagrama de classe, devido ao facto de as classes não estarem todas inseridas num único diagrama, podendo uma determinada classe participar em vários diagramas.

Nos diagramas de classes existem três elementos de modelação que importam referir: classe, relação e multiplicidade.

- ❖ Classe:

Uma classe define os atributos e os métodos (operações) de um conjunto de objectos que compartilham o mesmo comportamento e possuem o mesmo conjunto de atributos (diz-se que todos estes objectos são instâncias da classe a que pertencem). As classes são representadas por rectângulos divididos em três partes, a primeira designa o nome da classe, a segunda os seus atributos e a terceira as operações da mesma (Figura 9).

Nome da classe
Atributos
Operações

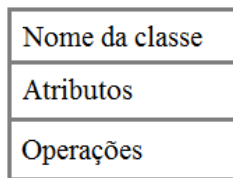



Figura 9 Diagrama de classes: representação genérica de uma Classe [2]

- Relação:

Uma relação, assim como o nome indica, representa uma relação entre diferentes objectos. Estas são representadas nos diagramas de classes por um traço e normalmente caracterizadas por um nome e se necessário pelo papel que os objectos apresentam nessas relações. As relações possíveis entre os objectos incluem: associação, dependência, herança, agregação e composição. Na Figura 10 apresenta-se a representação gráfica de cada um destes tipos de relações.

A associação é o mecanismo que permite que os objectos comuniquem entre si, descrevendo a conexão entre diferentes classes. Estas podem ter uma regra que especifica o seu propósito e ser uni ou bidireccional, indicando se apenas um deles pode enviar mensagens para o outro ou se os dois objectos participantes do relacionamento podem enviar mensagens entre si, respectivamente. A dependência trata-se de uma relação na qual uma mudança na especificação de um elemento pode alterar a especificação do elemento dependente. A dependência entre classes indica que os objectos de uma classe utilizam serviços dos objectos de outra classe. Uma relação de herança verifica-se quando uma classe herda todos os atributos e operações de uma outra classe, podendo sobrepor ou modificar algumas delas, assim como adicionar mais atributos ou operações próprias.

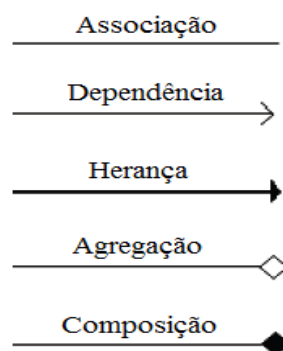



Figura 10 Diagrama de classes: representação gráfica dos diferentes tipos de relações entre objectos [2]

A agregação é um tipo especial de associação na qual as duas classes participantes não possuem um nível igual, porém fazem parte de um relacionamento "todo/parte", onde o objecto definido como "parte" é um atribuído do objecto definido como "todo" e onde o objecto "parte" é criado apenas se o objecto "todo" ao qual está agregado também for criado. A composição, à semelhança da agregação, forma um relacionamento "todo/parte", porém este relacionamento é "tão forte" que as "partes" não podem existir independentes, ou seja, apenas existem dentro do "todo", sendo destruídas caso o "todo" também o seja.

❖ Multiplicidade:

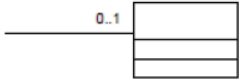
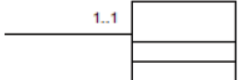
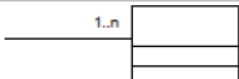
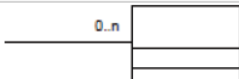
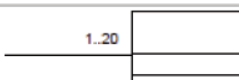
As relações são caracterizadas por possuírem uma multiplicidade, que assinala o número de objectos que participam numa determinada relação. A multiplicidade pode assumir diferentes formas. Na Figura 11 são apresentadas as formas mais comuns de multiplicidade e a simbologia associada a cada uma delas.

Multiplicidade	Simbologia
Zero ou um	0..1
Um para um	1..1
Um para muitos	1..n
Zero ou muitos	0..n
Um a x (entre um intervalo estabelecido)	1..20

Figura 11 Diagrama de classes: representação das formas mais comuns de multiplicidade [2]

2.3.3. DIAGRAMA ENTIDADE-RELAÇÃO

Os diagramas Entidade-Relação (também denominados por diagramas Entidade-Associação) são utilizados como ferramenta de auxílio à concepção de bases de dados. Estes vêm acompanhados por uma técnica de representação gráfica que ajuda na visualização das relações entre as diferentes entidades (conceitos) de um sistema de informação, no entanto, não fornece informações acerca das funções que manipulam os dados. Um diagrama Entidade-Relação (DER) é composto por três elementos principais: entidades, atributos e relações.

❖ Entidades:

Uma entidade pode ser definida como uma colecção ou conjunto de objectos do mundo real, sejam concretos (por exemplo: um computador ou robô) ou abstractos (por exemplo: um curso universitário), cujas informações são de interesse para o sistema. Ao serem identificadas as diferentes entidades que compõem um sistema é facilitado o processo de construção de uma base de dados, uma vez que são identificados os objectos relevantes para o sistema, e as suas características ou propriedades. Na elaboração de um DER, as entidades são representadas por rectângulos, como demonstra a Figura 12 (por exemplo a entidade "Proprietário").

❖ Atributos:

Na recolha e armazenamento das informações relacionadas com as entidades, estas informações podem ser agrupadas por categorias, isto é, por aquilo a que se designa de atributos. Os atributos de cada entidade descrevem as suas propriedades. Nos diagramas Entidade-Relação, os atributos aparecem num "compartilhamento" diferente da entidade a que pertencem, como por exemplo: o atributo "nome" da entidade "Proprietário", da Figura 12. Na Tabela 2 apresentam-se os principais tipos de atributos que se podem encontrar nos DER.

Tabela 2 Diagramas Entidade-Relação: tipos de atributos

Tipo de atributo:	Descrição:	Exemplo (Figura 12):
Atributo simples	Um único atributo representa uma característica.	Atributo "nome" da entidade "Proprietário".
Atributo concatenado	Vários atributos representam uma característica.	Atributos "marca", "modelo" e "matrícula" da entidade "Automóvel".
Chave primária	Um atributo que é sempre único dentro de uma entidade (não se repete).	Atributo "matrícula" da entidade "Automóvel".
Chave estrangeira	Atributo que se identifica com a chave primária de uma outra entidade.	Atributo "matrícula" da entidade "Marcação" relaciona-se com a chave primária da entidade "Automóvel".

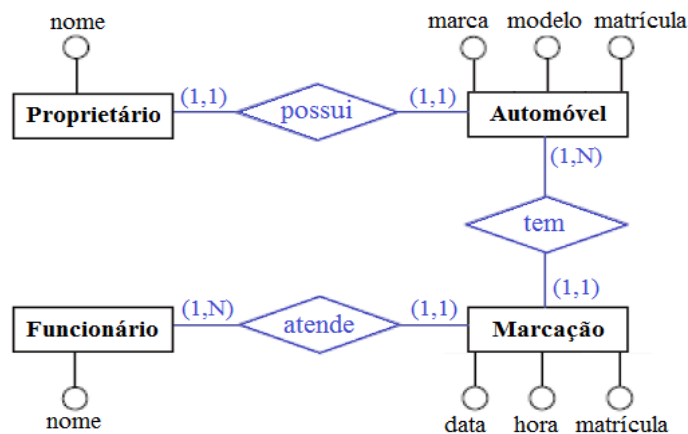



Figura 12 Exemplo de um Diagrama Entidade-Relação

- Relações:

Uma relação define uma associação ou ligação entre duas entidades. As relações são representadas nos DER através de losangos (por exemplo "possui" da Figura 12) ou outro símbolo semelhante a estes, como elipses, hexágonos, etc. É possível estabelecer diferentes tipos de relações entre entidades, podem ser do tipo: um para um, um para muitos ou muitos para muitos.

Uma relação é do tipo um para um, se a uma ocorrência de uma entidade corresponder uma e uma só ocorrência de uma outra entidade (exemplo: a cada entidade "Proprietário" corresponde uma e uma só ocorrência na entidade "Automóvel" da Figura 12). Uma relação é do tipo um para muitos, se a uma ocorrência de uma entidade corresponder uma ou mais ocorrências de uma outra entidade, sendo que a esta segunda apenas pode corresponder uma ocorrência da primeira. Na Figura 12 pode-se visualizar uma relação do tipo um para muitos, em que cada "Marcação" é atendida por um e um só "Funcionário" e cada "Funcionário" atende a uma ou várias "Marcações". Uma relação é do tipo muitos para muitos, se a uma ocorrência de uma entidade corresponder uma ou mais ocorrências de uma outra entidade.

2.4. Linguagens de programação Client Side

As linguagens de programação *client side* são linguagens executadas no lado do cliente, ou seja, no computador do próprio utilizador, e por isso são utilizadas nas situações em que a linguagem *server side* (executadas no lado do servidor) não tem alcance. Nas linguagens interpretadas na máquina cliente (*client side*) o conteúdo gerado é exibido conforme os

recursos disponíveis em cada navegador, ou seja, caso o navegador não tenha os recursos que estão nos *scripts* (códigos que fazem parte dos arquivos existentes num *website*), o utilizador será privado de visualizar parte do seu conteúdo. Por sua vez nas linguagens interpretadas num servidor (*server side*) a apresentação do conteúdo de uma página depende do servidor que a interpreta, assim quando um utilizador efectua uma requisição (ou pedido), o servidor processa os *scripts* que compõem a página e devolve ao cliente apenas o resultado na forma de HTML.

A seguir serão apresentadas algumas das linguagens de programação *client side*, amplamente utilizadas na construção de *websites*, nomeadamente HTML (*HyperText Markup Language*), CSS (*Cascading Style Sheets*) e JavaScript. Não é objectivo descrever exaustivamente todo o potencial de cada uma destas tecnologias, mas sim apresentar uma breve descrição, de forma a possibilitar ao leitor uma maior familiarização com estas tecnologias, e por consequência, uma melhor compreensão da interface desenvolvida.

2.4.1. HTML

O HTML surgiu no ano 1990 por intermédio do cientista britânico Tim Berners-Lee, com o objectivo de colmatar a inexistência de uma linguagem específica para a organização, quer lógica, quer artística, das informações agrupadas num determinado local da *Web* (o acesso aos dados era possível, mas não às informações ou páginas de informações relacionadas). A finalidade inicial era possibilitar a troca de informações e o acesso a documentos de pesquisas entre cientistas de diferentes universidades, porém o sucesso deste projecto foi tal que este esteve na base dos fundamentos da Internet, tal como a conhecemos actualmente [11].

HTML é o acrónimo de *HyperText Markup Language*, em português Linguagem de Marcação de Hipertexto, porém não se trata de uma linguagem de programação propriamente dita, mas de uma linguagem de formatação que define um conjunto de etiquetas (*tags*) que afectam a forma pela qual o texto, as imagens e restantes elementos de uma página são exibidos pelo navegador.

Os documentos HTML podem ser criados a partir de qualquer editor de texto, como por exemplo o Bloco de notas dos sistemas *Windows* (Figura 13), visto tratarem-se de ficheiros de texto simples. Existem outras linguagens mais avançadas que podem ser utilizadas no

desenvolvimento de páginas *Web*, no entanto, é difícil encontrar um *site* que não utilize a linguagem HTML, sendo até mais comum encontrar sites que utilizem apenas o HTML [2].

Na Figura 13 é possível visualizar a estrutura básica de um documento HTML. Esta é composta por um cabeçalho e pelo corpo do documento. A primeira linha do documento apresentado nesta figura indica que este se trata de um documento HTML. As *tags* *<html>* e *</html>* marcam o início e o fim deste código, respectivamente. O cabeçalho, identificado pelas *tags* *<head>* e *</head>*, contém informação relativa ao documento, como por exemplo o seu título (texto apresentado na barra superior do navegador) que é identificado pelas *tags* *<title>* e *</title>*, palavras-chave que possam ser úteis nos motores de busca, e outros dados que não são considerados como parte do conteúdo do documento. O corpo do documento é identificado *tags* *<body>* e *</body>*, e é o local onde é incluído todo o conteúdo de um *site*, habitualmente constituído por texto, imagens, ligações, entre outros elementos.

Para programar em HTML, não é necessário instalar *software* específico, porém, existem alguns *softwares* que facilitam o seu desenvolvimento através do uso de várias ferramentas combinadas, o que permite que sejam desenvolvidos com maior facilidade páginas *Web*. Um destes *softwares* com maior utilização é o *Macromedia Dreamweaver* da Adobe Systems [1], que como será mostrado posteriormente, no capítulo 4, foi o *software* usado para a implementação deste projecto.

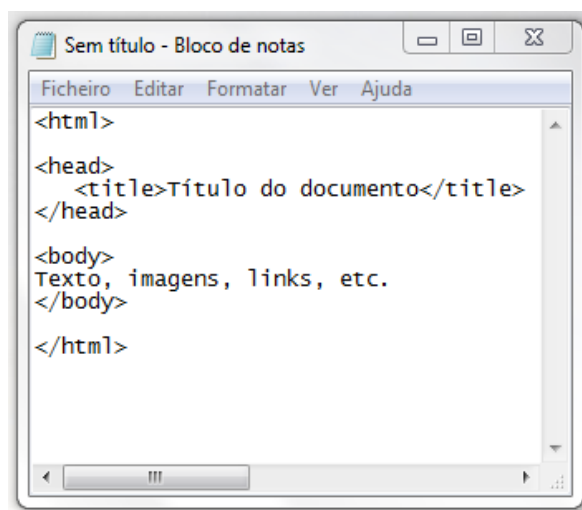



Figura 13 Estrutura básica de um documento HTML [2]

2.4.2. CSS

O CSS foi apresentado em Outubro de 1994 por Hakon Lie, com o intuito de facilitar a programação de *websites*, tarefa esta muito mais complexa nessa época. Para se chegar a um resultado simples, como a criação de uma tabela, os desenvolvedores de *websites* tinham necessidade de utilizar mais códigos face aos utilizados actualmente. A primeira versão do CSS, CSS1, surgiu em 1995, como projecto de um grupo de empresas do ramo da informática, conhecido por W3C [10]. Esta alcançou enorme destaque nos anos seguintes (entre 1997 e 1999), ficando conhecida por uma grande parte dos programadores.

O CSS é uma linguagem que determina a aparência (*layout*) de páginas *Web*, através da especificação de um conjunto de atributos de formatação da página, de preferências tipográficas e outras características do dispositivo cliente, de forma a garantir a continuidade visual do *site*. Esta é utilizada por milhares de programadores espalhados por todo o mundo e permite controlar as opções de margem, posicionamento, cores, altura e largura, estilos de linhas e imagens, sem necessidade de programar em HTML.

A sintaxe da linguagem CSS é um pouco diferente de sintaxe HTML, porém possibilita a alteração de quase todas as *tags* da linguagem HTML. A principal limitação desta linguagem reside na falta de reconhecimento de algumas versões de navegadores, sendo utilizada, portanto, como complemento ao HTML.

As principais vantagens da utilização do CSS são: facilidade de criação de *layouts*, que não necessitam de códigos muito complexos (existem bastantes “códigos prontos” que podem ser facilmente descarregados da Internet), com um único documento CSS é possível controlar vários documentos HTML e a rapidez de aprendizagem da mesma. Na Figura 14 apresenta-se, na sua forma mais elementar, a sintaxe da linguagem CSS, que como é possível constatar encontra-se dividida em três partes: *selector*, *propriedade* e *valor*. O *selector* indica o elemento/*tag* HTML ao qual será aplicada a propriedade (por exemplo: “*<body>*”), a *propriedade* designa o atributo que se deseja alterar (por exemplo: *font*, *color*, *background*, etc.) e o valor especifica a característica a ser assumida pela propriedade (por exemplo: tipo de letra “arial”, cor da letra verde, etc.).

selector { propriedade: valor }

Figura 14 Sintaxe da linguagem CSS

2.4.3. JAVASCRIPT

JavaScript foi originalmente desenvolvido por Brendan Eich da *Netscape Communications Corporation* sob o nome de Mocha, posteriormente teve o seu nome mudado para LiveScript e por fim JavaScript. JavaScript é uma linguagem interpretada embutida dentro de arquivos HTML e cujo funcionamento é bastante simples: sempre que um destes arquivos é carregado, o navegador interpreta o *script* e realiza as operações especificadas.

Embora seja bastante limitada (não suporta aplicações complexas nem transacções que necessitem de comunicação a partir da rede) a linguagem JavaScript, ao contrário da HTML que é uma linguagem estática, permite a implementação de animações com textos e imagens e diversas outras interacções com o utilizador, sendo assim considerada um "acessório" da HTML.

Por ser o JavaScript uma das poucas linguagens suportadas pela maioria dos navegadores mais populares, esta tornou-se uma linguagem alvo para muitos *frameworks* desenvolvidos em outras linguagens. Não obstante as limitações de desempenho inerentes à sua natureza dinâmica, a crescente velocidade das implementações JavaScript tem feito da mesma uma linguagem prática e de utilização massiva.

2.5. LINGUAGENS DE PROGRAMAÇÃO PARA PÁGINAS DINÂMICAS

Inicialmente as interfaces *Web* não eram mais do que documentos HTML, com texto e imagens interligados por *links* que permitiam navegar entre documentos. A necessidade de se utilizar a *Web* como um ambiente de base para aplicações mais complexas como, por exemplo transacções com bases de dados, motivou o desenvolvimento de novas tecnologias, nomeadamente de tecnologias para programação de páginas dinâmicas.

Softwares servidores de páginas dinâmicas como o *Active Server Pages* (ASP), *Java Server Pages* (JSP) e *Page Hypertext Preprocessor* (PHP) têm sido amplamente utilizados para o desenvolvimento de interfaces dinâmicas geradas a partir de uma consulta à base de dados e para o desenvolvimento dinâmico que se faz pelo aproveitamento de um ou mais arquivos em diversas interfaces usando directivas de *Server Side Include* (SSI).

A seguir serão apresentadas as tecnologias para criação de páginas dinâmicas ASP, JSP e PHP. Tal como nas linguagens de programação *client side*, também aqui não é pretendido descrever exaustivamente todo o seu potencial, mas uma breve descrição das mesmas.

2.5.1. ASP

A ASP, criada pela Microsoft [27], trata-se de uma tecnologia orientada a objectos, utilizada para desenvolver aplicações interactivas na *Web*, ou seja, cujo conteúdo é dinâmico (vai sendo alterado através da interacção com o utilizador). Utilizada por mais de 2 milhões de *sites* em todo o mundo [42], esta não é na realidade uma linguagem, mas sim uma estrutura de bibliotecas básicas para processamento de linguagens de *script* (por exemplo: VBScript, JScript, PerlScript ou Python). Estes *scripts* são interpretados no lado do servidor e o que é enviado para o lado do utilizador (por exemplo, para o navegador) é apenas a saída que habitualmente corresponde a uma linguagem de marcação como o HTML, o que permite que qualquer navegador existente no mercado seja capaz de aceder a um *site* que utilize esta tecnologia.

As páginas ASP respeitam uma arquitectura de 3 níveis, como apresentado na Figura 15. Esta arquitectura permite que um servidor (nível 2) possa servir de intermediário entre o navegador instalado no lado do cliente (nível 1) e uma base de dados (nível 3), pelo acesso transparente ao mesmo através da tecnologia ADO (*ActiveX Data Objects*), que fornece os elementos necessários para a conexão com o sistema de gestão da base de dados.

As principais vantagens da utilização da tecnologia ASP incluem: a independência do navegador, a segurança do código fonte, o suporte de diferentes linguagens e a possibilidade de interacção com bases de dados. Como os *scripts* usados nesta tecnologia são interpretados no lado do servidor, esta é independente do navegador usado, podendo ser acedidos a partir de qualquer um destes. Da mesma forma, ao retornar apenas o resultado é preservado o código fonte. Esta permite ainda visualizar, actualizar e/ou adicionar dados num sistema de gestão de bases de dados.

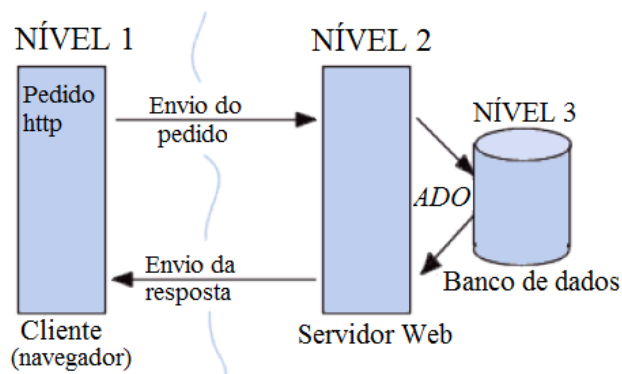



Figura 15 Arquitectura das páginas dinâmicas ASP (*Active Server Pages*)

2.5.2. JSP

A linguagem JSP, criada pela *Sun Microsystems* (subsidiária da Oracle Corporation) [29], baseia-se na linguagem de programação Java e tem como propósito simplificar o processo de desenvolvimento dinâmico de *websites*. O seu funcionamento é similar ao da tecnologia ASP, sendo os comandos processados pelo servidor antes da página HTML ser enviada para o navegador, como demonstra a Figura 16. Como se pode visualizar na referida figura, em primeiro lugar o cliente envia um pedido, através do navegador pela Internet, de uma página ao servidor. O servidor *Web* reconhece o pedido especial (extensão .jsp) e envia-o para o componente que o manipula, recuperando dados de uma base de dados ou de outro tipo de repositório, e envia a resposta de volta para o servidor. Por fim, o servidor envia a página já formatada de volta para o cliente, que a pode visualizar no seu navegador.

Por ser baseada na linguagem Java, a tecnologia JSP tem a vantagem da portabilidade de plataforma, que permite a sua execução em diversos sistemas operacionais, como o Windows e o Linux [36]. Esta tecnologia permite ao desenvolvedor *Web* produzir aplicações que acedam a bases de dados, recolham informações a partir de formulários e manipulem arquivos no formato de texto.

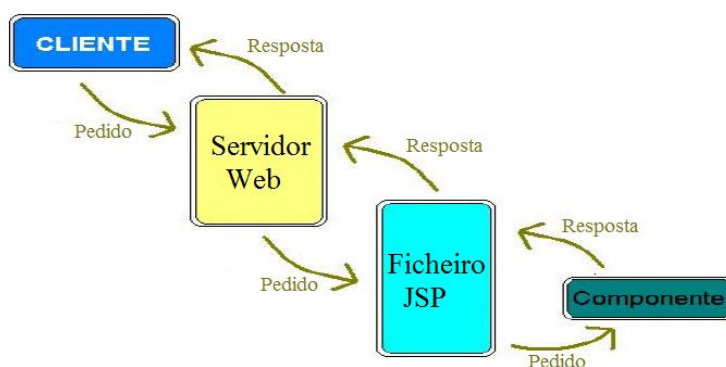



Figura 16 Arquitectura das páginas dinâmicas JSP (*Java Server Pages*) [36]

2.5.3. PHP

O PHP apresenta-se como uma alternativa gratuita e de código aberto (*open source*) ao ASP e JSP, e constitui-se como uma linguagem de *scripts* bastante utilizada no desenvolvimento de aplicações *Web*, podendo ser incorporada dentro do código HTML. Esta tecnologia surgiu em 1994, sob o nome de *Personal Home Page*, como um projecto pessoal de Ramus Lerdof com o intuito de controlar os acessos à sua página *Web*. Actualmente esta linguagem é conhecida por *Page Hypertext Preprocessor* e é controlada pela *Apache Software Foundation* [31].

A linguagem PHP baseia-se nas linguagens de programação C, Java e Perl, podendo ser vista como uma combinação de linguagens de programação e servidores de aplicações. Tal como nas linguagens anteriores (ASP e JSP), os comandos são processados pelo servidor antes de a página ser enviada para o navegador.

Uma das características mais importantes desta tecnologia é o suporte de um número elevado de sistemas de gestão de base de dados, como o mSQL (ou mini SQL), mySQL, PostgreSQL, Oracle, dBase, Interbase, Sybase, entre outros [40]. Para além disso, esta suporta ainda um grande número de outros serviços, através do uso de protocolos como o SNMP (*Simple Network Management Protocol*), POP3 (*Post Office Protocol*), IMAP (*Internet Message Access Protocol*), NNTP (*Network News Transfer Protocol*) e HTTP (*HyperText Transfer Protocol*).

O PHP é o líder de mercado de servidores *Web* gratuitos e está presente em cerca de 55% da *World Wide Web* [40]. Este é compatível com várias plataformas, o que significa que este é executado, no seu formato original, em diferentes versões dos sistemas operativos UNIX e Windows. O PHP está actualmente na versão 5, chamada de PHP5, na sua versão estável 5.5.5. [31]. As várias versões do PHP foram sendo aclamadas e premiadas nos últimos anos. O PHP3 foi finalista em 1999 no LinuxWorld Editor's Choice Awards, na categoria de bibliotecas/ferramentas de programação, tendo alcançado o segundo lugar (sendo apenas ultrapassado pelo ColdFusion). Também a combinação PHP3/MySQL alcançou o prémio de sistema de gestão de bases de dados do ano em 1998.

No extracto de código que se segue é apresentado um pequeno exemplo do funcionamento de um arquivo PHP (identificado pelas *tags* *<?php* e *?>*) embutida no meio de uma página HTML (identificada pelas *tags* *<html>* e *</html>*).

```
<html>
<head>
    <title> Exemplo de código php </title>
</head>
<body>
<?php
    $a = "Olá Mundo!";
    echo $a;
?>
</body>
</html>
```

O código PHP acima apresentado resume-se na definição de uma variável (variável *a*, identificada por *$a*) e respectiva impressão para o ecrã (navegador usado pelo cliente). Neste é possível observar alguns aspectos relativos à sintaxe básica. O primeiro é que em cada linha de código tem de haver um ponto e vírgula, de forma a indicar ao sistema o fim da instrução (exemplo: *echo $a;*). O segundo reside no facto de em PHP todas as variáveis serem identificadas pelo caracter *$* e não ser necessário defini-las antes de serem utilizadas nem definir um tipo para ela (*int*, *float*, *char*).

3. PLATAFORMA *WEB* - MODELAÇÃO

Neste capítulo é apresentada a fase de projecto para o desenvolvimento da plataforma *web*. Inicialmente é introduzido o conceito do projecto e quais os objectivos que se pretendem alcançar com a realização do mesmo. Numa fase seguinte será apresentada a análise dos requisitos necessários para atender às necessidades da plataforma, de forma a permitir reunir as ideias necessárias para a projecção da mesma. Por último, serão apresentados os diagramas UML (*Unified Modeling Language*) relativos ao projecto desenvolvido, dos quais fazem parte o diagrama de classes e o diagrama de *use cases;* e o diagrama entidade-relação (também denominado por modelo E-R), do modelo relacional de bases de dados. Estes diagramas têm por finalidade expressar numa linguagem modelo todas as funcionalidades e relações que se pretendem atribuir à plataforma a ser desenvolvida.

3.1. INTRODUÇÃO

O desenvolvimento deste trabalho de dissertação teve como principal objectivo a obtenção de competências na área de desenvolvimento de uma página *web* com recurso a bases de dados.

Foram utilizadas as linguagens HTML e PHP, já apresentadas anteriormente nas subsecções 2.4.1 e 2.5.3 do capítulo 2, respectivamente. O HTML foi utilizado para o *layout* e disposição de todo o conteúdo visível da página e o PHP para a execução dos *scripts* que permitem a interacção com as diversas funcionalidades do *website*. Foi ainda utilizada a linguagem de programação *Javascript* numa fase mais avançada do trabalho, para o *design* da página e para inclusão de efeitos visuais.

Para interacção com as bases de dados criadas para o *website* utilizou-se a linguagem SQL, de forma a providenciar um conjunto diversificado de serviços de acesso aos dados nelas contidos, tais como: definição, manipulação e consulta de dados.

A página *web* criada destina-se a ser utilizada por pessoas com interesse pela área dos automóveis. Pretende-se que a página apresente uma interface gráfica apelativa e ao mesmo tempo simples, de forma a cativar possíveis visitantes.

Com o intuito de proporcionar aos utilizadores do *website* um papel mais activo e simultaneamente mais dinâmico, este disponibiliza uma série de recursos, tais como: notícias acerca dos recentes desenvolvimentos no que diz respeito ao mundo automóvel, informação técnica de algumas das marcas de automóveis com maior índice de vendas, um fórum em que cada utilizador pode (após efectuar o registo obrigatório) criar novos tópicos de debate e consultar/comentar os tópicos existentes, galeria de imagens e fotos, entre outros.

3.2. ANÁLISE DE REQUISITOS

Após terem sido definidos os objectivos propostos para a elaboração desta dissertação seguiu-se a elaboração da análise dos requisitos necessários para a criação do *site*. Para isso foi realizada uma análise a *sites* relacionados com o tema deste trabalho [4] [5] [30] [35] [41]. Após a análise realizada decidiu-se implementar os seguintes requisitos:

- qualquer utilizador pode consultar informações relativas a cada marca automóvel (descrição histórica, modelos, *concept cars* e *links* úteis), glossário, novidades/inovação, lista de *sites* úteis, contactos e pesquisar cada um dos diferentes modelos de automóvel que constam na base de dados criada, através da barra de pesquisa;

- apenas os utilizadores registados podem aceder ao fórum;
- qualquer utilizador pode visualizar/comentar as notícias e aceder à galeria de imagens e fotos;
- após efectuar o *login* os utilizadores registados podem responder ao questionário;
- o administrador é responsável por toda a gestão do *site* (registo dos utilizadores, fórum, notícias, galeria e questionário) e pela criação e gestão das contas nas redes sociais.

Para satisfazer alguns destes requisitos torna-se necessário efectuar o armazenamento dos dados, nomeadamente dos dados pessoais de acesso dos utilizadores, de forma a ser possível aceder a estes para consulta e/ou escrita de novos dados e consequente controlo de acesso a algumas das funcionalidades do *site*.

O *site* pode ser acedido por três tipos de utilizadores: visitante, utilizador registado e administrador. Por visitante (ou utilizador "normal") entende-se todo o utilizador que navega pelo *site* sem ter ainda efectuado o registo no mesmo. Após ter efectuado o registo no *site* passa a ser considerado como utilizador registado. O administrador é aquele que tem a responsabilidade de planear, organizar/alocar os recursos e controlar todos os serviços da plataforma *web*. De forma a esclarecer o papel de cada um destes tipos de utilizadores no *site* serão apresentadas a seguir as funcionalidades/tarefas a cada um deles associadas:

Utilizador "normal" (visitante):

Pode consultar a informação relativa a cada marca, glossário, novidades/inovações, *sites* úteis, contactos e a classificação (individual e por equipas) do campeonato de Fórmula 1 da presente época; pesquisar cada um dos modelos de automóveis que constam da base de dados com recurso à barra de pesquisa; visualizar e comentar notícias; aceder à galeria de fotos e vídeos e às redes sociais (*facebook*, *twitter* e *youtube*); efectuar o registo e sugerir possíveis alterações de forma a melhorar o *site* desenvolvido.

Utilizador registado:

Para além das funcionalidades afectas aos visitantes, após efectuar o registo os utilizadores podem: efectuar o login no *site*; consultar e alterar os seus dados do perfil (por exemplo, alterar a sua *password* de registo); aceder ao fórum e responder ao questionário *online*.

Administrador:

Para além das funcionalidades afectas aos utilizadores registados o administrador pode aceder às estatísticas do *site* e tem por tarefa gerir o fórum, as contas nas redes sociais, o questionário *online* (inserir, alterar ou eliminar questões) e os registos dos utilizadores através do painel de controlo (visualizar contas inactivas durante um intervalo de tempo seleccionado, com possibilidade de apagar essas contas ou enviar um *e-mail* a relembrar os utilizadores que podem aceder ao *site*); e adicionar novas fotos, vídeos e notícias.

3.3. DIAGRAMA DE *USE CASES*

O diagrama de *Use Cases* (em português, diagrama de casos de uso) tem por objectivo descrever as funcionalidades do sistema do ponto de vista dos seus utilizadores e identificar os diferentes actores que interagem com o mesmo.

Tal como já foi referido no capítulo anterior, nomeadamente na subsecção 2.3.1, o diagrama de *Use Cases* é representado por actores, *use cases* e relacionamentos entre estes elementos. Os actores correspondem aos utilizadores do sistema e podem ser utilizadores humanos ou um outro sistema. No caso do *site* desenvolvido neste projecto este é composto por três actores: visitante, utilizador registado e administrador.

Após serem identificados os actores do sistema podem então ser identificados os *use cases*, isto é, as funcionalidades que cada um destes actores encontra à sua disposição. As funcionalidades associadas ao sistema desenvolvido neste projecto encontram-se representadas na Tabela 3.

Tabela 3 *Use Cases* (funcionalidades) do sistema elaborado

Use Case	Descrição
Explorar o *site*	Pesquisar e consultar informação, e efectuar comentários às notícias.
Gestão	Gerir a página, conteúdos e as contas dos utilizadores.
Acesso exclusivo	Aceder ao fórum e questionário *online*.
Registo no *site*	Acessível a todos os visitantes que o desejem fazer.
Controlo de acesso	Restringir a utilização do *site* por parte de alguns utilizadores.

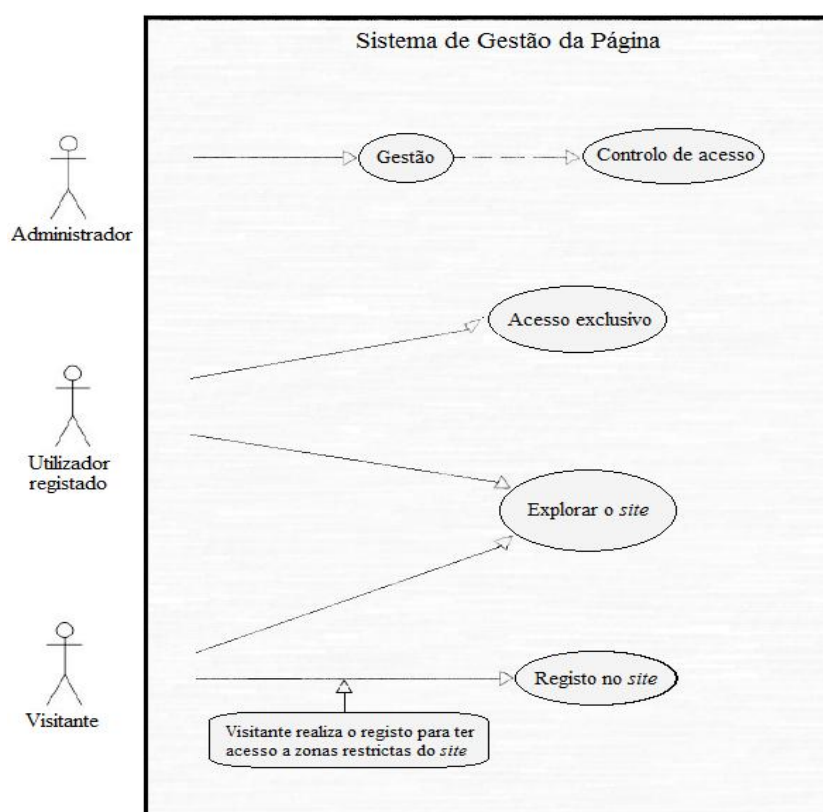



Figura 17 Diagrama de *use cases* da plataforma *Web*

Definidos que estão os actores e as funcionalidades associadas ao sistema pode então ser representado o diagrama de *use cases*, onde será apresentada a forma como estes se relacionam entre si. O diagrama referido encontra-se ilustrado na Figura 17.

Analisando a Figura 17 pode-se constatar que os visitantes podem explorar o *site* e efectuar o registo no mesmo. Após efectuar o registo passa a ser considerado como utilizador registado onde para além de poder continuar a explorar o *site* pode ainda aceder a zonas exclusivas (“acesso exclusivo”). O administrador do sistema tem por funcionalidades a sua gestão e o controlo de acesso a ele.

3.4. DIAGRAMA DE CLASSES

Após terem sido identificados os diferentes actores que irão interagir com o sistema desenvolvido e descritas as funcionalidades associadas a cada um deles pode então ser construído o diagrama de classes do sistema. O objectivo do diagrama de classes consiste em descrever os vários tipos de objectos (entidades e conceitos) do sistema, as características comuns em termos de propriedades (atributos) e o relacionamento entre eles. Os elementos principais deste tipo de diagramas são: classe, atributos, operações e multiplicidade.

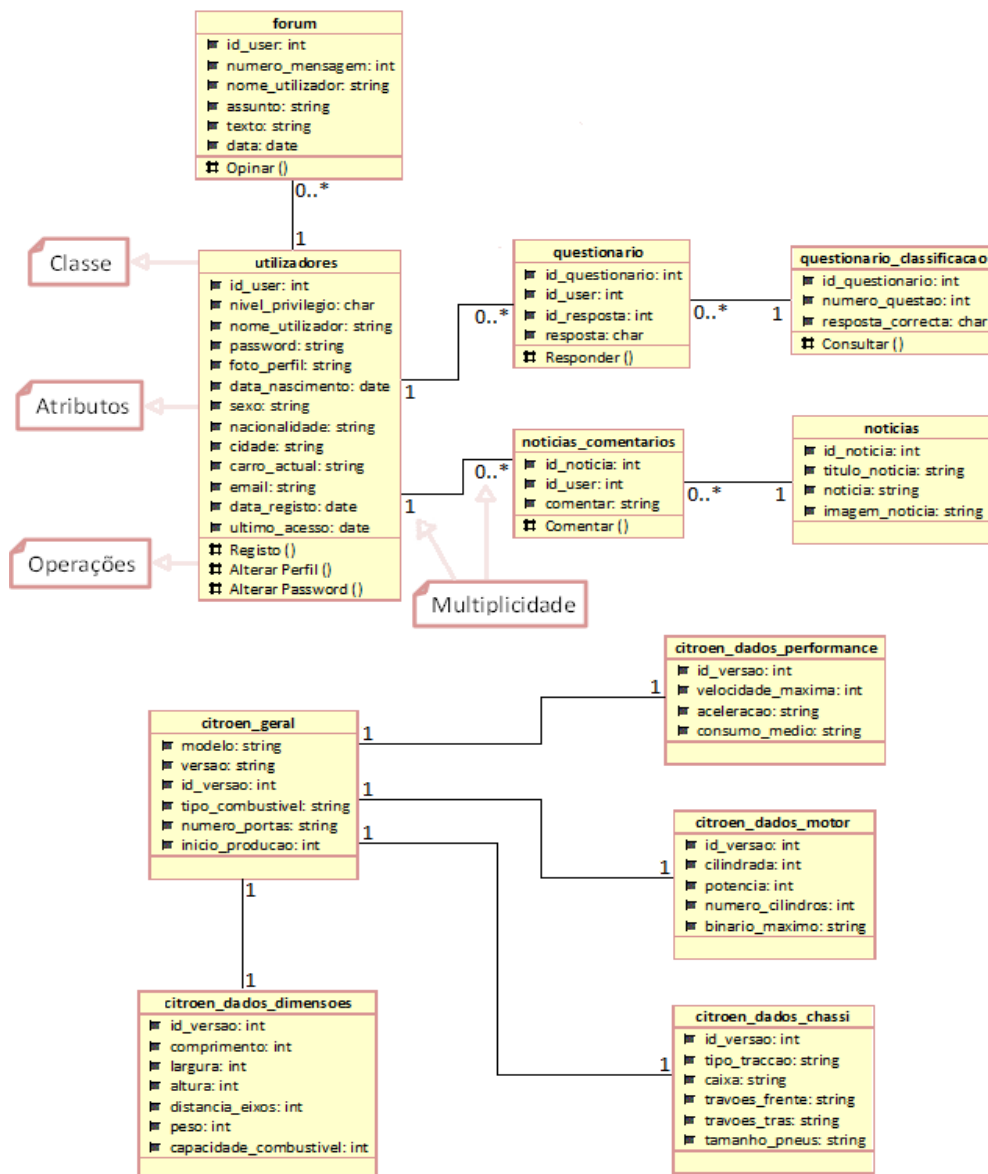



Figura 18 Diagrama de classes do *site* desenvolvido

Na Figura 18 apresenta-se o diagrama de classes da plataforma desenvolvida, onde é possível observar os diferentes elementos descritos. Através desta é possível ter uma ideia de como se relacionam as diferentes tabelas presentes na base de dados criada e do modo como são geridos e relacionados os dados nelas contidos. Na tabela "utilizadores" encontram-se os dados referentes a cada um dos utilizadores registados no *site*. Cada um destes utilizadores pode efectuar vários comentários nas notícias ("noticias_comentarios") e responder ao questionário ("questionario") as vezes que pretender, desta forma verifica-se uma relação de 1 ("1") para muitos ("0..*"). A tabela "forum" armazena os dados das mensagens escritas pelos utilizadores no fórum, podendo cada um dos utilizadores escrever várias mensagens, o que corresponde a uma relação de 1 para muitos. As tabelas referentes aos comentários das notícias e do questionário contêm um utilizador por comentário e por questionário realizado, logo uma relação de muitos para 1.

Nas tabelas referentes às marcas de automóveis (Citroën, Ford, Peugeot, Renault e Seat) verificam-se relações de 1 para 1, uma vez que existe uma única versão de automóvel (identificada pelo atributo "id_versao") para cada um dos conjuntos de dados ("marca_dados_performance", "marca_dados_motor", "marca_dados_chassi" e "marca_dados_dimensoes"). Importa referir que não foram incluídas no diagrama de classes apresentado na Figura 18 todas as tabelas da base de dados criada, uma vez que esta possui 35 tabelas, e que levariam a que o diagrama a apresentar se tornasse muito extenso. Assim, no que diz respeito às tabelas referentes às marcas de automóveis foram incluídas apenas as tabelas de apenas uma das marcas, neste caso, da Citroën. Para as restantes marcas as tabelas apresentam o mesmo funcionamento mudando apenas o nome das classes.

3.5. DIAGRAMA ENTIDADE-RELAÇÃO

Após construído o diagrama de classes pode então ser elaborado o diagrama entidade-relação (DER), com vista a auxiliar o processo de concepção da base de dados. Os DER, à semelhança dos diagramas de classes, são também constituídos pelos elementos: classes, atributos e relações. No entanto, os DER são baseados numa técnica de representação gráfica que auxilia a visualização das relações entre as diferentes entidades de uma base de dados.

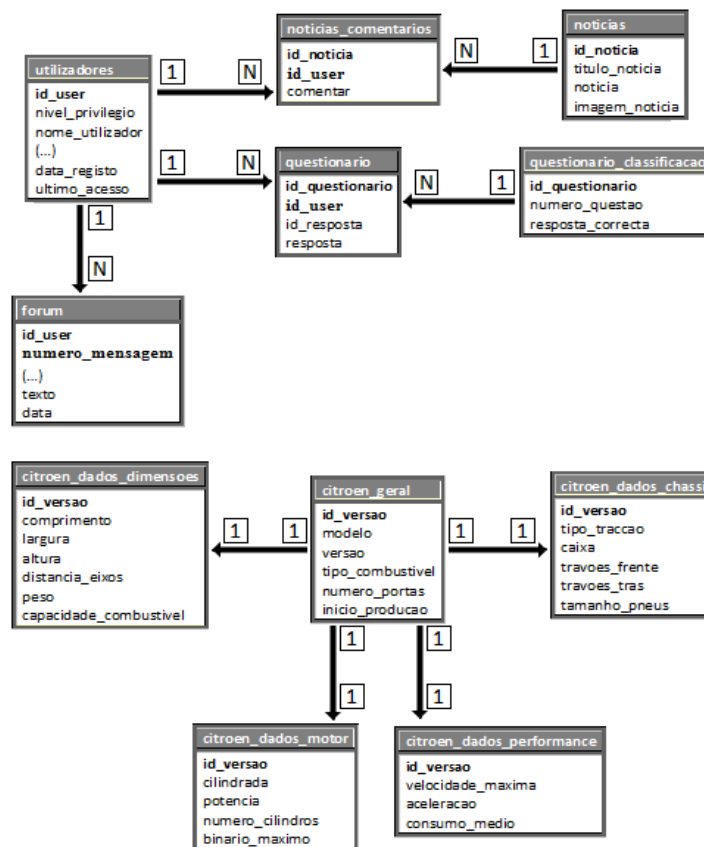



Figura 19 Diagrama entidade-relação da plataforma *Web*

No DER representado na Figura 19, no qual é possível observar o relacionamento entre as diferentes tabelas da base de dados, isto é, o modo como estas se relacionam entre si, encontrando-se assinalado a negrito as chaves primárias de cada uma destas tabelas. As chaves primárias, do ponto de vista de uma base de dados relacional, referem-se a conjuntos de um ou mais campos de uma tabela, cujos valores nunca se repetem e que podem ser usadas como índice para os demais campos dessa mesma tabela.

As relações entre as tabelas já foram explicadas na secção anterior (3.4) portanto não serão alvo de estudo nesta secção sob pena de se repetir informação. No que diz respeito às chaves primárias pode-se constatar que a tabela "utilizadores" tem como chave primária o campo "id_user" que não pode ter dois ou mais registos com o mesmo valor, o que permite identificar inequivocamente cada utilizador. A tabela "noticias" tem como chave primária o campo "id_noticia" que permite identificar cada uma das notícias presentes na base de dados. Visto que cada utilizador pode efectuar vários comentários na mesma notícia ou em notícias diferentes a tabela "noticias_comentarios" apresenta uma chave primária composta, ou seja, formada por mais de um campo, nomeadamente, pelos campos

“id_noticia” e “id_user”, o que permite associar cada comentário em determinada notícia ao utilizador responsável pelo mesmo.

A tabela “forum” à semelhança da tabela anterior também possui uma chave primária composta formada pelos campos “id_user” e “numero_mensagem” que permite identificar o utilizador que publicou determinada mensagem colocada no fórum. Visto se tratar de uma chave primária composta os valores de cada um destes campos podem se repetir, porém a combinação destes valores não pode ser repetida, ou seja, o utilizador com um “id_user = 1” apenas pode ter associada uma única entrada na tabela de dados com “numero_mensagem = 24”. Contudo pode existir uma outra entrada na tabela com “numero_mensagem = 24” desde que o valor do campo “id_user” seja diferente de 1, ou seja, referente a outro utilizador.

As tabelas “citroen_geral”, “citroen_dados_performance”, “citroen_dados_motor”, “citroen_dados_chassi” e “citroen_dados_dimensoes” apresentam uma chave primária simples formada pelo campo “id_versao” pois existe apenas uma entrada em cada uma destas tabelas para cada versão de determinado automóvel. Assim, torna-se simples associar cada conjunto de dados a cada versão de determinado modelo automóvel.

3.6. DESENVOLVIMENTO DA BASE DE DADOS

Após ter sido efectuada a análise de requisitos e elaborados os respectivos diagramas UML (*Unified Modeling Language*) tornou-se possível concretizar a implementação e estruturação das tabelas que compõem a base de dados que será utilizada para armazenar os dados que serão apresentados no *site* e/ou inseridos a partir do mesmo.

A base de dados foi criada directamente no servidor de alojamento através da aplicação *phpMyAdmin*. Esta aplicação permite criar e remover bases de dados, criar, remover e alterar tabelas, inserir, remover e editar campos, e definir as chaves primárias, sem necessidade de inserir códigos SQL (*Structured Query Language*); sendo apenas necessário especificar os nomes das tabelas e dos campos, assim como definir os seus tipos de dados, tamanhos e chaves primárias. O *phpMyAdmin* gera o código SQL e cria automaticamente as tabelas e/ou os campos que o utilizador especificou.

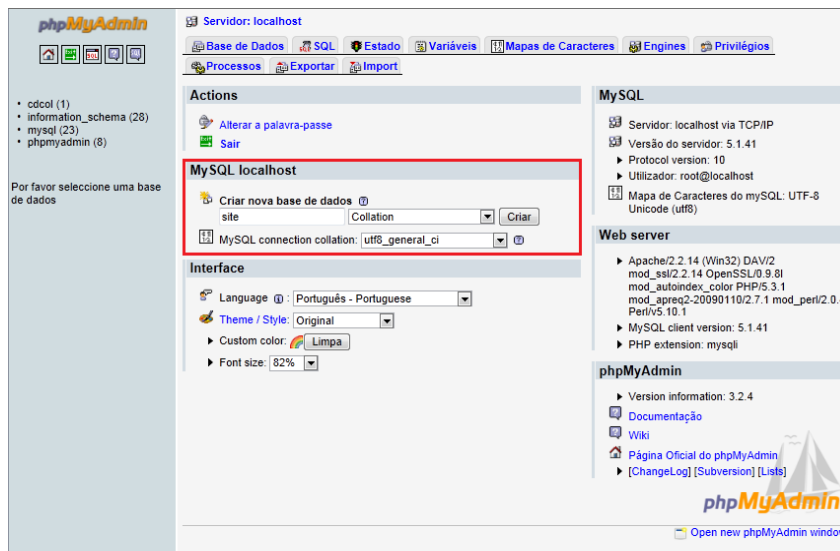



Figura 20 Janela principal da aplicação *phpMyAdmin*

Esta aplicação pode ser descarregada gratuitamente da Internet através do *site* da *phpMyAdmin* [33]. Após a instalação da aplicação e ao executar a mesma é iniciada uma janela igual à apresentada na Figura 20.

Na janela principal do *phpMyAdmin* é possível seleccionar qual a base de dados que se pretende aceder através do menu lateral (apresentado à esquerda do leitor) ou através do separador "Base de Dados" localizado no menu central no topo da janela. Uma vez que se trata do primeiro acesso à aplicação são apenas apresentadas as tabelas já incluídas no ficheiro de instalação da mesma, neste caso: "cdcol", "information_schema", "mysql" e "phpmyadmin". Assim sendo foi criada uma nova base de dados à qual foi atribuído o nome "site", conforme se pode constatar no rectângulo vermelho assinalado na Figura 20. Esta base de dados será usada para armazenamento dos dados que serão exibidos no *site* desenvolvido e/ou inseridos a partir deste.

3.6.1. CRIAÇÃO DAS TABELAS DE DADOS

Após criada a base de dados podem ser definidas as tabelas que a constituem, tal como apresentado na Figura 21. A base de dados é composta por 35 tabelas, conforme referido anteriormente na secção 3.4. De forma a clarificar o modo como estas podem ser criadas e quais os parâmetros que devem ser configurados serão expostos a seguir os passos efectuados para criação de uma destas tabelas, em particular da tabela "citroen_geral". Durante este processo é necessário definir apenas dois parâmetros: o parâmetro "Nome" que tal como o nome indica permite associar à tabela um nome (exemplo: "citroen_geral") e o parâmetro "Number of fields" que designa o número de campos da tabela (exemplo: "6").

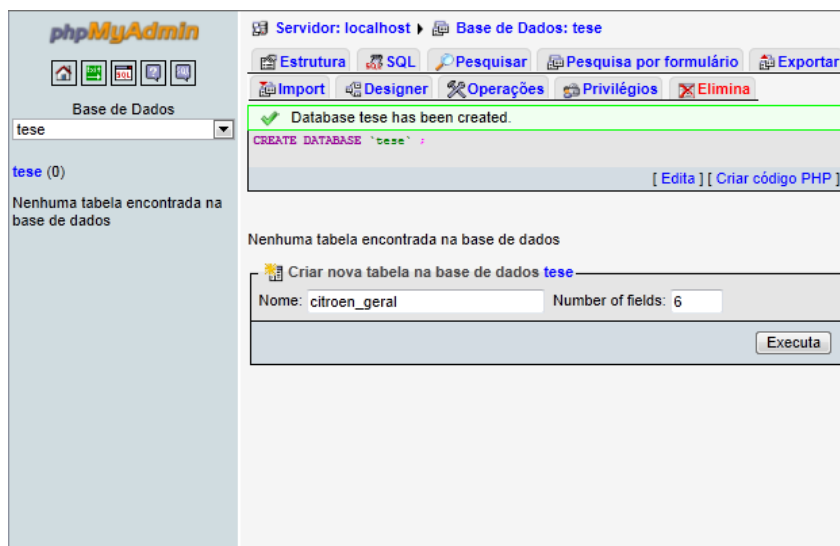



Figura 21 *phpMyAdmin* – criação de uma nova tabela na base de dados

A tabela "citroen_geral", já apresentada no diagrama Entidade-Relação elaborado (Figura 19 – secção 3.5) é composta por 6 campos, entre eles: "id_versao", "modelo", "versao", "tipo_combustivel", "numero_portas" e "inicio_producao". Quando executada a criação da tabela surge uma nova janela, semelhante à da Figura 22, onde são definidos e parametrizados cada um destes campos. O significado de cada um destes campos será explicado com maior detalhe no capítulo da implementação do projecto.

O tipo de dados usado por cada campo é definido no parâmetro "Tipo" e o tamanho desses dados em "Tamanho/Valores", apresentados na Figura 22. Os dados podem ser do tipo *int*, *varchar*, *text* ou *date*. Os dados do tipo *int* correspondem a números inteiros, como por exemplo: 1, 2 e 3. Os dados do tipo *varchar* representam *strings* de tamanho variável de tamanho entre 0 e 255 (exemplos: C1, C2 e C3). Os dados do tipo *text* permitem armazenar grandes quantidades de texto, geralmente usados para guardar mensagens de *e-mail* na base de dados e por sua vez os dados do tipo *date* para armazenar datas, de forma a ser possível manipular as datas através da base de dados. Após terem sido definidos e parametrizados os campos, o processo de criação da tabela termina. Na Figura 23 apresenta-se o aspecto final desta tabela.

Figura 22 *phpMyAdmin* – definição e parametrização dos campos da tabela criada

Figura 23 *phpMyAdmin* – estrutura da tabela criada

Após ter sido criada a tabela é então possível preencher a mesma com os dados que se pretendem armazenar. A cada conjunto de dados definidos (preenchimento de todos os campos) dá-se o nome de registos. Na Figura 24 apresenta-se um exemplo do modo de inserção de um novo registo, onde é possível observar que existe um separador próprio para este efeito denominado por "Insere", que se encontra assinalado na figura dentro de um rectângulo com contorno vermelho.

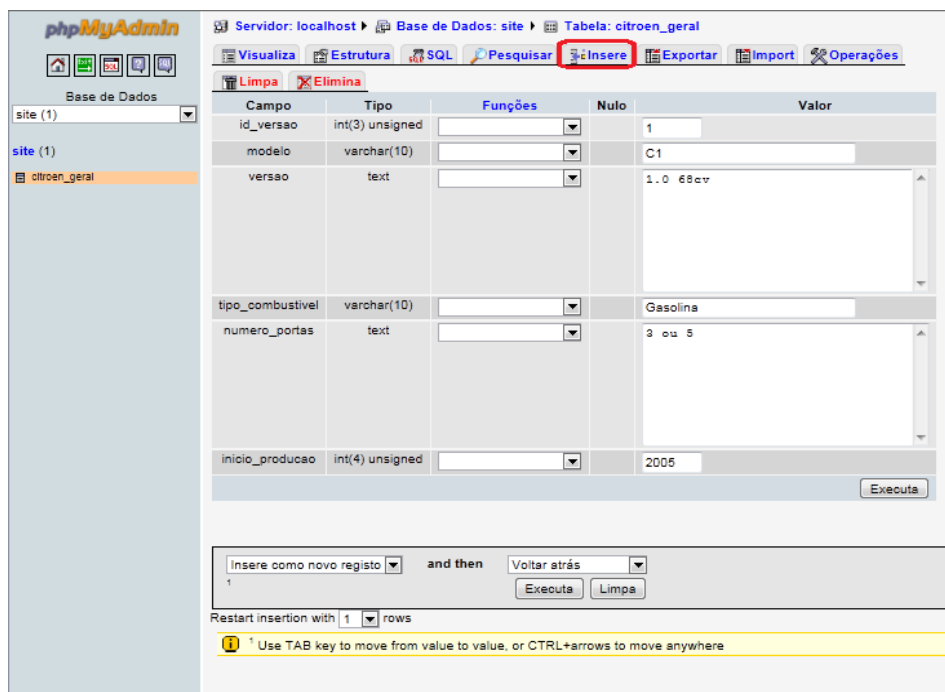



Figura 24 ***phpMyAdmin*** **– inserir novo registo**

Depois de preenchidos todos os campos do registo e após ser executado o comando, isto é, após ser pressionado o botão “Executa”, é possível visualizar que o registo foi adicionado com sucesso à tabela criada, conforme apresentado na Figura 25. Para adicionar mais registos à tabela de dados devem ser seguidos os mesmos passos efectuados para a inserção do registo dado como exemplo, sendo apenas alterados os valores atribuídos a cada um dos campos da tabela.

Chegado a este ponto e com o conhecimento adquirido acerca do funcionamento do *phpMyAdmin* tornou-se possível criar todas as tabelas necessárias, bem como os respectivos registos, para o desenvolvimento do projecto. Estas encontram-se representadas na Figura 26.

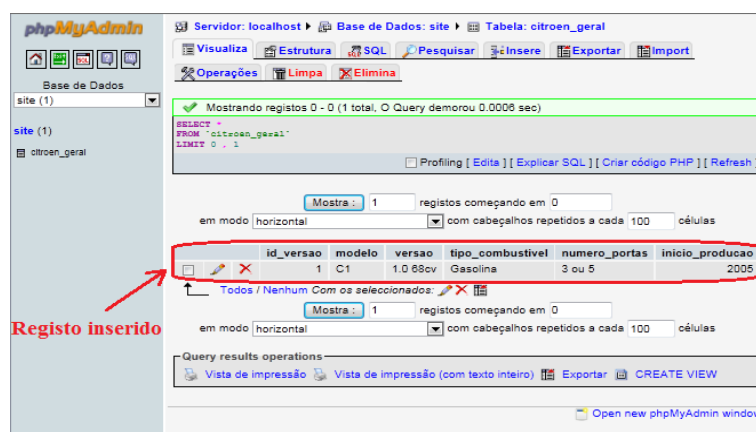



Figura 25 ***phpMyAdmin*** **– registo inserido com sucesso na tabela de dados**

Servidor: localhost ▸ Base de Dados: site

Estrutura | SQL | Pesquisar | Pesquisa por formulário | Exportar | Import | Designer | Operações | Privilégios | Elimina

Tabela	Acções	Registos	Tipo	Collation	Tamanho	Suspenso
citroen_dados_chassi		16	MyISAM	latin1_swedish_ci	3.4 KB	-
citroen_dados_dimensoes		16	MyISAM	latin1_swedish_ci	2.8 KB	-
citroen_dados_motor		16	MyISAM	latin1_swedish_ci	2.8 KB	-
citroen_dados_performance		16	MyISAM	latin1_swedish_ci	2.6 KB	-
citroen_geral		16	MyISAM	latin1_swedish_ci	2.7 KB	-
ford_dados_chassi		16	MyISAM	latin1_swedish_ci	3.5 KB	-
ford_dados_dimensoes		16	MyISAM	latin1_swedish_ci	2.8 KB	-
ford_dados_motor		16	MyISAM	latin1_swedish_ci	2.9 KB	-
ford_dados_performance		16	MyISAM	latin1_swedish_ci	2.7 KB	-
ford_geral		16	MyISAM	latin1_swedish_ci	2.8 KB	-
formula1_classificacao_equipas		11	MyISAM	latin1_swedish_ci	2.3 KB	-
formula1_classificacao_individual		33	MyISAM	latin1_swedish_ci	2.6 KB	-
forum		17	MyISAM	latin1_swedish_ci	1.6 KB	-
fotos		54	MyISAM	latin1_swedish_ci	5.2 KB	-
noticias		4	MyISAM	latin1_swedish_ci	3.5 KB	-
noticias_comentarios		6	MyISAM	latin1_swedish_ci	2.2 KB	-
peugeot_dados_chassi		20	MyISAM	latin1_swedish_ci	3.8 KB	-
peugeot_dados_dimensoes		20	MyISAM	latin1_swedish_ci	3.0 KB	-
peugeot_dados_motor		20	MyISAM	latin1_swedish_ci	3.1 KB	-
peugeot_dados_performance		20	MyISAM	latin1_swedish_ci	2.8 KB	-
peugeot_geral		20	MyISAM	latin1_swedish_ci	2.9 KB	-
questionario		94	MyISAM	latin1_swedish_ci	3.9 KB	-
questionario_classificacao		25	MyISAM	latin1_swedish_ci	3.0 KB	-
renault_dados_chassi		22	MyISAM	latin1_swedish_ci	4.1 KB	-
renault_dados_dimensoes		22	MyISAM	latin1_swedish_ci	3.1 KB	-
renault_dados_motor		22	MyISAM	latin1_swedish_ci	3.2 KB	-
renault_dados_performance		22	MyISAM	latin1_swedish_ci	2.9 KB	-
renault_geral		22	MyISAM	latin1_swedish_ci	3.0 KB	-
seat_dados_chassi		24	MyISAM	latin1_swedish_ci	4.3 KB	-
seat_dados_dimensoes		24	MyISAM	latin1_swedish_ci	3.3 KB	-
seat_dados_motor		24	MyISAM	latin1_swedish_ci	3.4 KB	-
seat_dados_performance		22	MyISAM	latin1_swedish_ci	2.9 KB	-
seat_geral		24	MyISAM	latin1_swedish_ci	3.1 KB	-
utilizadores		4	MyISAM	latin1_swedish_ci	6.6 KB	-
videos		15	MyISAM	latin1_swedish_ci	2.9 KB	-
35 tabela(s)	Soma	751	MyISAM	latin1_swedish_ci	111.6 KB	0 Bytes

Figura 26 ***phpMyAdmin*** **– tabelas da base de dados criada**

3.6.2. INTERACÇÃO COM A BASE DE DADOS

A interacção com a base de dados engloba o seu acesso, ou seja, a consulta, inserção ou até substituição dos seus dados. Este tipo de interacção é realizado, geralmente, na linguagem de programação PHP, no código da página respectiva. Na Figura 27 é fornecido um exemplo de um código PHP (identificado pelas *tags* *<?php* e *?>*) que permite ligar com o servidor e com a base de dados, necessários para conseguir interagir com determinada base de dados.

Para efectuar a ligação com o servidor em que o *site* se encontra alojado utiliza-se a função *mysql_connect*. Esta possui como argumentos o nome do servidor (*$Local_serve*), o nome do utilizador que pretende aceder ao mesmo (*$usuario_serve*) e a respectiva *password* de acesso (*$senha_serve*). Se não for possível ligar com o servidor é retornada uma mensagem de aviso "O servidor não responde!". Após efectuada a ligação com o servidor, pode ser realizada a ligação com a base de dados. Para tal, é utilizada a função *mysql_select_db*, que selecciona, através da ligação ao servidor efectuada anteriormente (*$Ligacao_servidor*), a base de dados que se encontra no primeiro argumento da função (*$banco_de_dados*). Também aqui é apresentada, em caso de falha da ligação, uma mensagem de aviso com o conteúdo "Não foi possível conectar-se à base de dados!".

```php
<?php

$local_serve = "localhost";          // localização do servidor
$usuario_serve = "root";             // nome do utilizador
$senha_serve = "*********";          // senha
$banco_de_dados = "site";            // nome do banco de dados

// ligação com o servidor
$ligacao_servidor = @mysql_connect($local_serve,$usuario_serve,$senha_serve) or die ("O servidor não responde!");

// conectar-se ao banco de dados
$ligacao_base = @mysql_select_db($banco_de_dados,$ligacao_servidor) or die ("Não foi possível conectar-se à base de dados!");
```

Figura 27 Código em PHP para ligação com o servidor e com a base de dados

4. PLATAFORMA *WEB* - IMPLEMENTAÇÃO

Neste capítulo serão apresentadas as páginas existentes no *site* (também conhecido por sítio, em português), bem como uma explicação das potencialidades de cada uma delas e do modo como se relacionam entre si. Serão também apresentadas todas as opções tomadas (tecnologias escolhidas) ao longo do projecto e o modo como foram integradas no sítio à medida que este foi evoluindo.

De forma a possibilitar uma melhor compreensão do funcionamento do sítio, entendeu-se necessária a "divisão" deste projecto em quatro secções: página principal, perfil de utilizador, página de registo e barra de menu central.

O sítio foi implementado através do *software Macromedia Dreamweaver*, que pode ser descarregado via *web* a partir do *site* da companhia norte-americana Adobe Systems [1]. Trata-se de um *software* de desenvolvimento voltado para aplicações *web*, de fácil utilização e configuração, e que suporta diversas linguagens de programação, tais como: CSS (*Cascading Style Sheets*), JavaScript, Ajax (*Asynchronous Javascript and XML*), PHP e JSP (*JavaServer Pages*). Com excepção da linguagem JSP, foram utilizadas todas estas

linguagens durante o processo de criação do sítio, conforme será possível atestar ao longo das próximas secções.

O *Dreamweaver* possibilitou liberdade de escolha em relação à construção do sítio, no que diz respeito à estrutura das suas páginas, que foram implementadas através do uso de folhas de estilo em HTML (*HyperText Markup Language*), e à programação necessária para a interacção dos utilizadores com o *site*, como o acesso à base de dados.

4.1. PÁGINA PRINCIPAL

A página principal do *site* desenvolvido (AutoInfo), representada na Figura 28, corresponde à página apresentada ao utilizador quando este acede à raiz do *site*, localizado em *http://localhost/tese/index.php*. Esta página é constituída essencialmente por uma barra superior, logótipo do *site*, coluna da esquerda, coluna da direita, barra do menu central, conteúdo central e rodapé.

A barra superior contempla uma *drop down menu* [14] que permite pesquisar através de palavras-chave alguns dos conteúdos do sítio, nomeadamente marcas de automóveis e respectivos modelos. Numa segunda secção desta barra são apresentados alguns ícones que dão acesso às páginas das redes sociais criadas para o sítio, nomeadamente Facebook, Twitter e Youtube. Esta possui ainda um botão que permite realizar o *login* no sítio.

O logótipo, localizado abaixo da barra superior refere-se à forma particular como o sítio é representado graficamente, pela escolha de um desenho com uma tipografia específica, que pretende facilitar a identificação do mesmo perante o seu "público-alvo".

Na coluna da esquerda é incluído um separador que permite ao utilizador aceder a determinados conteúdos do sítio, que variam consoante o tipo de utilizador, isto é, visitante, utilizador registado ou administrador (esta distinção já foi abordada no capítulo 3, aquando da análise de requisitos e do diagrama de *use cases*, secções 3.2 e 3.3, respectivamente). É ainda nesta coluna que se encontram localizados o calendário e o relógio para apresentação da data e das horas, respectivamente. Por último, é apresentado um *script* desenvolvido em JavaScript com notícias, testes e comparativos e fotos do mundo automóvel em constante actualização.

A barra de menu central possibilita aos utilizadores acederem a todas as páginas que compõem o sítio. Mais à frente serão apresentadas todas as páginas que estes poderão consultar, através da referida barra (secção 4.4).

Na secção de conteúdo central apresenta-se o conteúdo da página acedida, nomeadamente as notícias principais, um quadro com uma mensagem de "boas vindas" ao *site* e dois *scripts*, um com notícias sobre o desporto automóvel actualizadas via *Google Web Elements* [19] e um outro com informação relativa ao preço médio dos combustíveis, actualizada de forma automática e constante.

A coluna da direita apresenta informação relativa ao campeonato de Fórmula 1 época 2012/2013, nomeadamente as classificações individual e por equipas. O rodapé inclui uma lista com alguns dos *links* mais comuns que se podem encontrar em qualquer *website*, de forma a facilitar o seu acesso, bem como os respectivos direitos do autor.

A seguir serão apresentadas as potencialidades de cada um dos aspectos referidos.

Figura 28   Página principal do sítio desenvolvido (AutoInfo)

4.1.1. Barra superior, logótipo e rodapé

O desenvolvimento da página principal do sítio foi iniciado pela criação da barra superior, do logótipo e do rodapé do mesmo. No que diz respeito à barra superior esta corresponde na realidade a duas barras distintas, uma apresentada caso o utilizador não tenha realizado o *login* no *site* e outra caso tenha sido efectuado o *login*, como é representado na Figura 29.

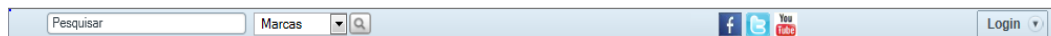



Figura 29 Página principal – aspecto da barra superior

Para verificar se o utilizador se encontra autenticado, testa-se se este tem sessão iniciada, ou seja, se já efectuou o *login* no sítio. Para tal, é necessário recorrer ao código que a seguir se apresenta, código este que se encontra na página principal.

```
<?php session_start(); ?>
…
<body>
<?php
  if ($_SESSION['id'] == "")
  {
      require("barra_superior.html");
  }
  else
  {
      require("login/barra_superior.php");
  }
?>
…
</body>
```

Neste extracto de código é possível observar que em primeiro lugar é iniciada a sessão, através da função *session_start*, e em seguida verificado o *id* da sessão. Caso o *id* esteja em branco, isto é, caso o utilizador ainda não tenha efectuado o *login*, é chamado o ficheiro *barra_superior.html*, e a barra superior apresenta o aspecto da primeira barra apresentada na Figura 29. Caso o utilizador tenha realizado o *login*, é então atribuído um *id* à sessão e requerido o ficheiro respectivo, *barra_superior.php*. Neste caso, a barra superior apresenta o aspecto da segunda barra apresentada na figura referida.

Em ambos os casos (duas barras) pode ser encontrada numa primeira secção um *drop down menu* onde se encontram os campos de pesquisa, com possibilidade de seleccionar o que se pretende pesquisar, seja marcas de automóveis ou respectivos modelos. Ao iniciar a escrita na caixa de texto surgem sugestões de pesquisa, tanto de marcas como de modelos, de

forma a facilitar a pesquisa, como demonstra a Figura 30. Estas sugestões foram implementadas em *jquery* e *php*, e efectuam a pesquisa nas tabelas de dados das marcas e modelo pelas palavras introduzidas na barra de pesquisa pelo utilizador.

Figura 30 Barra superior - secção de pesquisa de marcas e modelo

Se o utilizador efectuar a pesquisa baseada em marcas, será pesquisado nas tabelas de dados referentes às mesmas (*citroen_geral*, *ford_geral*, *peugeot_geral*, *renault_geral* e *seat_geral*) o conteúdo da pesquisa. Caso seja encontrada correspondência com uma marca contida na base de dados será então encaminhado para a página do sítio da marca pesquisada, que pode também ser acedido através do separador "Marcas" da barra de menu central conforme será exposto na subsecção 4.4.1. A pesquisa baseada nos modelos apresenta um funcionamento semelhante à pesquisa por marcas, sendo esta efectuada a partir das tabelas dos modelos.

Numa segunda secção da barra superior encontram-se os ícones de acesso às páginas das redes sociais, conforme *print screen* apresentado na Figura 31. Estas incluem uma conta no *Facebook* (acessível em *https://www.facebook.com/pages/Auto-Info/369620166499078*), outra no *Twitter* (acessível em *https://twitter.com/AutoInfo*) e outra no *Youtube* (acessível em *http://www.youtube.com/channel/UCYhU-ZCPoR5JVYDYN0kXJCA/feed*).

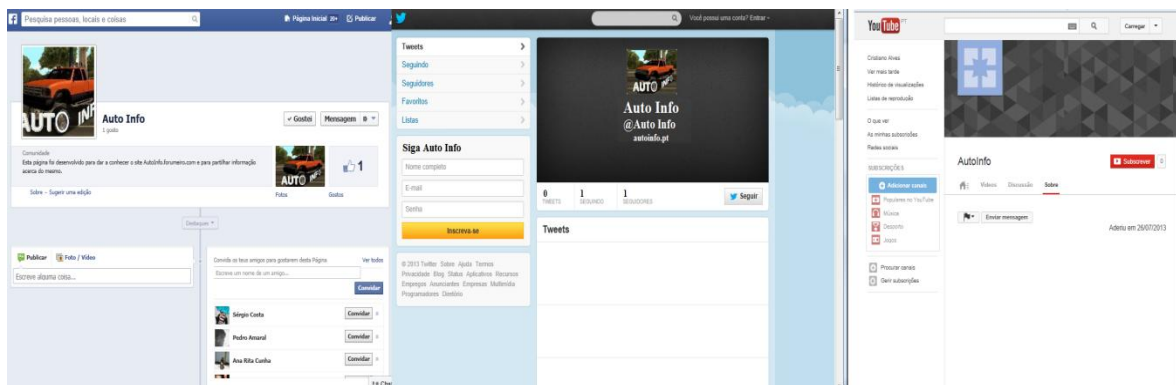



Figura 31 Contas nas redes sociais criadas para o sítio

Figura 32 Barra superior - secção de autenticação

Por fim, na última secção da barra superior encontra-se o botão que permite realizar o *login* no sítio. Para isso basta passar com o cursor do rato por cima da caixa de "*Login*", exibida na Figura 32, e surgirá então um formulário, desenvolvido com *javascript*, *css*, *html* e *php*, onde podem ser inseridos os dados para autenticação do utilizador.

Caso o utilizador já tenha efectuado a autenticação no sítio é apresentado em alternativa a mensagem "Bem-vindo: *nome de utilizador* (Sair)". Para sair da conta é apenas necessário clicar em "Sair", sendo então a sessão destruída através do uso da função própria para o efeito, ou seja, pela função *session_destroy*, conforme demonstra o extracto de código que a seguir se apresenta, sendo o utilizador encaminhado para a página principal "*index.php*".

```
<?php
    session_start();
    session_destroy();
    header("location:../index.php");
?>
```

Logo abaixo da barra superior é apresentado o logótipo que tal como já foi referido identifica o sítio perante o seu "público-alvo" através de uma representação gráfica particular. Esta representação gráfica pode ser visualizada na Figura 33.

Figura 33 Logótipo do sítio AutoInfo

Inicio | Contactos | Sugestões | Mapa do Site   Copyright © 2013 AutoInfo.pt Todos os direitos reservados

Figura 34   Rodapé do sítio AutoInfo

Através do rodapé do *site* é possível aceder a algumas das suas áreas de interesse, nomeadamente à página principal ("*Inicio*"), página de contactos ("*Contactos*"), página de sugestões ("*Sugestões*") ou mapa do site ("*Mapa do Site*"), como representado na Figura 34.

4.1.2. COLUNA DA ESQUERDA E COLUNA DA DIREITA

A coluna da esquerda inclui um separador que tem por objectivo facilitar o acesso à informação relativa ao perfil de utilizador, caso este tenha efectuado a autenticação no sítio. Se o acesso ao sítio tiver sido realizado por um visitante não é apresentada qualquer informação relativa ao perfil de utilizador, uma vez que este não efectuou a autenticação (seja por não possuir ainda um registo válido seja por não ter efectuado o *login* apesar de ter um registo válido). Nesta situação, representada na Figura 35 a), são somente apresentados dois *scripts*: um para apresentação da data e hora do servidor e um outro para exibição de notícias, testes e comparativos, e outras informações acerca do mundo automóvel.

Um utilizador ao efectuar a autenticação no *site* são-lhe apresentadas as funcionalidades acima referidas para os visitantes, porém com um menu adicional com informação relativa ao perfil desse utilizador, em concreto: o nome, idade, país, cidade, viatura actual e a foto de perfil, conforme registado na Figura 35 b). Este menu inclui ainda *links* para acesso a algumas páginas do sítio que eventualmente poderão ser de interesse para o utilizador, como o questionário *online*, o fórum e as páginas para editar o perfil ou alterar a *password* de registo. Mais à frente serão detalhados os aspectos referentes a este menu.

O administrador terá, em relação aos utilizadores registados, dois *links* extras para gestão e administração do *site*, representados por "Painel de Controlo" e imagem apresentada (quadrado branco com três barras vermelhas de diferentes tamanhos). Esta situação encontra-se representada pela Figura 35 c). À semelhança dos anteriores também estes *links* serão alvo de estudo nas próximas secções.

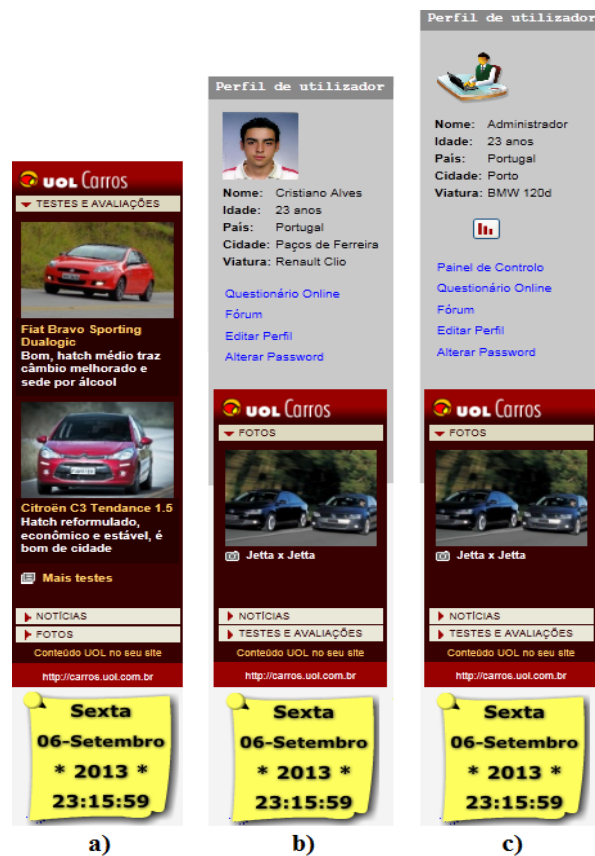



Figura 35 a) Coluna da esquerda para visitantes, b) Coluna da esquerda para utilizadores registados, c) Coluna da esquerda para o administrador

Pos.	Piloto	Pontos
1	Sebastian Vettel	157
2	Fernando Alonso	123
3	Kimi Raikkonen	116
4	Lewis Hamilton	99
5	Mark Webber	93
6	Nico Rosberg	84
7	Felipe Massa	57
8	Romain Grosjean	41
9	Paul di Resta	36
10	Jenson Button	33
11	Adrian Sutil	23
12	Sergio Perez	16
13	Jean-Eric Vergne	13
14	Daniel Ricciardo	11
15	Nico Hulkenberg	7
16	Pastor Maldonado	0
17	Valtteri Bottas	0
18	Esteban Gutierrez	0
19	Jules Bianchi	0
20	Charles Pic	0
21	Max Chilton	0
22	Giedo van der Garde	0

a)

Pos.	Equipa	Pontos
1	Red Bull Racing-Renault	250
2	Mercedes	183
3	Ferrari	180
4	Lotus-Renault	157
5	Force India-Mercedes	59
6	McLaren-Mercedes	49
7	STR-Ferrari	24
8	Sauber-Ferrari	7
9	Williams-Renault	0
10	Marussia-Cosworth	0
11	Caterham-Renault	0

b)

Figura 36 a) Coluna da direita: "*classificação individual*", b) Coluna da direita: "*classificação por equipas*"

A coluna da direita assim como já foi referido exibe informação relativa ao campeonato de Fórmula 1 da presente época (2012/2013), nomeadamente as classificações individual e por equipas, representadas na Figura 36 a) e b), respectivamente. A apresentação desta informação no sítio engloba a inclusão anterior da mesma na respectiva tabela da base de dados "*formula1_classificacao*", de forma a ser possível a sua consulta. Todo este processo resume-se ao código apresentado no Anexo A.

4.1.3. CONTEÚDO CENTRAL

A secção de conteúdo central engloba o conteúdo da página acedida. No que diz respeito à página principal do sítio esta inclui as notícias principais, uma mensagem de "boas vindas" ao *site* e dois *scripts*, um com notícias sobre o desporto automóvel e outro com informação relativa ao preço médio dos diferentes tipos de combustíveis, como demonstra a Figura 37.

As notícias principais da página principal estão constantemente a rodar, isto é, avançam para a notícia seguinte de forma automática e a uma velocidade constante (previamente definida), em forma de *slide*. Tal só se tornou possível uma vez que foi associado a estas notícias um easy slider para apresentação das mesmas. Foi ainda implementada uma outra tecnologia denominada por *shadowbox*, de forma a permitir abrir (ampliar) a notícia que se pretende consultar e efectuar (caso se pretenda) comentários às mesmas, como se demonstra na Figura 38. Ao abrir uma notícia é possível visualizar do lado esquerdo da *shadowbox* a imagem da notícia e os comentários a esta associados e do lado direito o seu título e conteúdo.

Estas duas soluções (*easy slider* e *shadowbox*) foram implementadas com recurso às tecnologias *jquery*, *css* e *html*. Por serem necessários conhecimentos acerca destas linguagens que não se enquadram nos objectivos definidos para este projecto, nomeadamente de *jquery*, foram descarregadas da internet duas soluções já desenvolvidas, uma para implementação do *easy slider* [13] e outra para implementação da *shadowbox* [39], às quais foram efectuadas as devidas alterações para conformidade com o *site* desenvolvido.

Figura 37 Página principal – conteúdo central

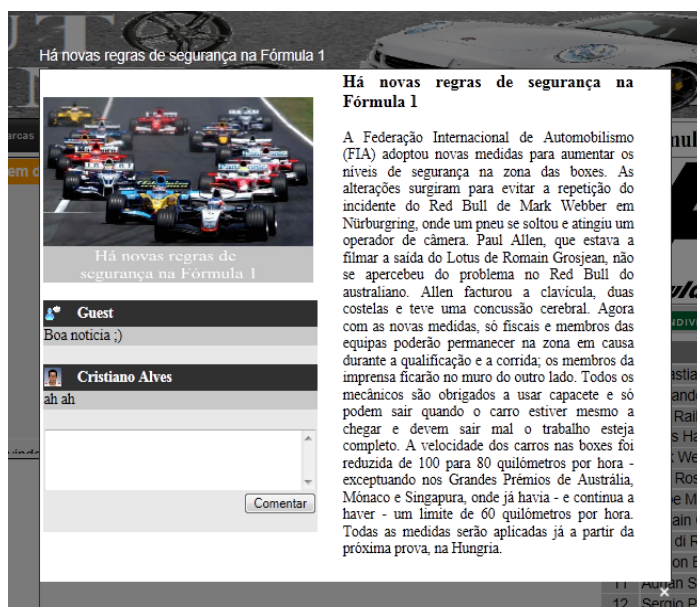



Figura 38 Página principal – notícia aberta através da *shadowbox*

Por baixo das notícias principais encontram-se dois *scripts*. O primeiro onde são apresentadas as notícias sobre o Desporto Automóvel a partir de uma API (*Application Programming Interface*) desenvolvida pela Google [19], em que são disponibilizadas notícias de diversos *sites* actualizadas frequentemente (por norma, diariamente). O segundo *script* abrange informação referente ao preço médio dos principais tipos de combustível automóvel em Portugal Continental e a variação do seu preço, nomeadamente gasolina, gasóleo e GPL (*gás de petróleo liquefeito*) auto. Esta informação é actualizada diariamente através do *site* Mais Gasolina que contém um directório interactivo de postos de abastecimento de Portugal com os preços actualizados dos combustíveis [26].

4.2. PÁGINA DE REGISTO

Uma das páginas mais importante do sítio desenvolvido corresponde à página de registo, uma vez que é através desta que os utilizadores, até aqui considerados como visitantes, têm acesso a algumas das suas secções. Assim, após efectuar o registo no sítio o utilizador é considerado como um "utilizador registado" passando a ter acesso às páginas de consulta e alteração dos dados de perfil, fórum e questionário *online* (após se autenticar no *site*).

A página de registo pode ser acedida através da opção ►***Efectuar registo*** do formulário de *login*, localizado na barra superior, ao clicar-se sobre o botão "*Login*", conforme apresentado na Figura 39. Surge então um formulário igual ao apresentado na Figura 40 onde devem ser introduzidos os dados para o registo, nomeadamente, o nome de utilizador, *password*, email, data de nascimento, cidade e país, sexo, carro actual e foto de perfil.

Figura 39 Formulário de login – efectuar registo

Figura 40 Página de registo

Alguns dos campos do formulário de registo são de carácter obrigatório, portanto caso não seja preenchido um destes campos não é possível efectuar o registo, sendo apresentada uma mensagem de aviso “Campos não preenchidos” e indicado quais os campos que ficaram por preencher, conforme apresentado na Figura 41.

Ao ser introduzido o nome de utilizador no formulário de registo, este é comparado com os nomes de utilizador já existentes na base de dados, de forma a verificar a disponibilidade do mesmo. Caso o nome já esteja em uso é apresentada uma mensagem de aviso “O nome *nome de utilizador* já está em uso”, como exibido na Figura 41. Para análise das *passwords* e correspondente indicação do nível de segurança foi desenvolvido um algoritmo específico, apresentado no Anexo B.

Após o visitante efectuar o registo, os dados preenchidos são enviados para a base de dados que contém os dados dos utilizadores, sendo a *password* encriptada pela função md5 [15] [22], que a transforma numa sequência de caracteres, como demonstra a Figura 42. Desta forma, é garantida maior segurança no tratamento dos dados de cada utilizador, em particular, da sua *password* de acesso.

Figura 41 Menu de registo: aviso da disponibilidade do nome de utilizador, nível de segurança da *password* e dados em falta

nome_utilizador	password	email	data_nascimento	pais	cidade	sexo	foto_perfil	carro_actual
Cristiano Alves	4f754eca115987590e1dc0814a889e16	carvalhocristiano@hotmail.com	1990-05-19	Portugal	Porto	Masculino	7fae3.jpg	Renault Clio

Figura 42 Registo dos dados do utilizador na base de dados

Em caso de esquecimento da *password* o utilizador pode solicitar o envio de uma nova *password* para o *email* com que efectuou o registo, através da opção ►***Recuperar password*** do formulário de *login*, já apresentado anteriormente na Figura 39. Ao clicar nesta opção é apresentada uma nova página onde o utilizador terá de introduzir o seu *email*, como mostra a Figura 43. Tal como referido anteriormente, quando o utilizador solicita a recuperação da sua *password* de acesso, esta é-lhe enviada para o seu email, como demonstra a Figura 44. No Anexo C encontra-se representado o algoritmo responsável por gerar aleatoriamente a nova *password*.

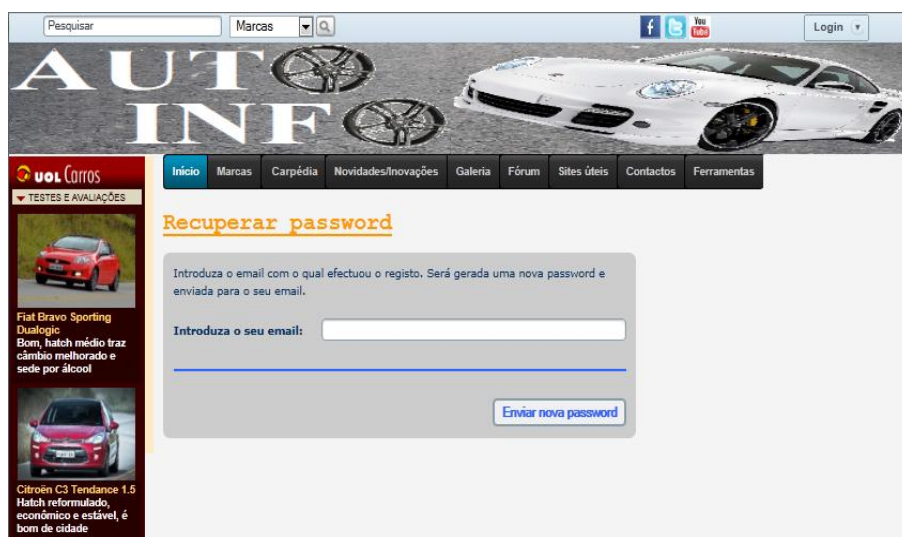



Figura 43 Página de recuperação de *password*

Figura 44 Envio da nova *password* para o email do utilizador

4.3. PERFIL DE UTILIZADOR

Como já foi referido anteriormente ao efectuar a autenticação no sítio é apresentado ao utilizador um separador com os dados de perfil referentes a esse utilizador. Neste separador, apelidado por "perfil de utilizador", pode-se visualizar o nome, idade, país, cidade e foto de perfil. Este inclui ainda alguns *links* para acesso a determinados páginas do sítio, como o questionário *online*, o fórum, e as páginas de edição do perfil e de alteração da *password*. Na Figura 45 é apresentado um exemplo de um perfil de utilizador, onde podem ser observadas todas estas funcionalidades.

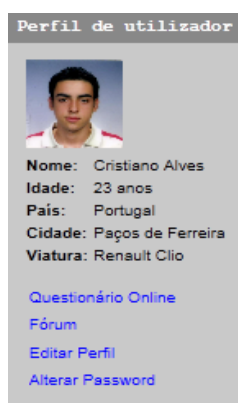



Figura 45 Exemplo de um perfil de utilizador

4.3.1. PAINEL DE CONTROLO

Apenas o administrador do sítio tem acesso ao painel de controlo, a partir do seu perfil de utilizador. Este acesso é efectuado através do *link* "Painel de Controlo" que o reencaminha para a página respectiva (Figura 46).

Figura 46 Perfil de utilizador do administrador do sítio

No painel de controlo, apresentado na Figura 47, o administrador pode verificar a data do registo e do último acesso ao sítio de todos os utilizadores e o tempo decorrido entre ambas. Através deste o utilizador pode ainda seleccionar as contas inactivas durante determinado período de tempo (seleccionado através da opção “Último acesso à”) e enviar um *email* a esses utilizadores a relembrar que podem usufruir dos serviços do sítio ou então caso prefira eliminar a(s) conta(s) desse(s) utilizador(es). Ao seleccionar a opção “Enviar emails” é enviado um *email* semelhante ao da Figura 48 para os utilizadores seleccionados.

Figura 47 Painel de controlo das contas dos utilizadores

Assunto: AutoInfo
De: Cristiano Alves <carvalhocristiano@hotmail.com>
Data: Hoje, 23:09:41 WEST
Para: carvalho.cristiano@sapo.pt

Olá Cristiano Alves

O **AutoInfo** relembra-o que ainda pode desfrutar dos nossos serviços. Caso pretenda cancelar o seu registo é favor enviar um email para: tese_0713@hotmail.com

Pode aceder ao nosso site através do link:
http://localhost/tese

Os melhores cumprimentos,
AutoInfo

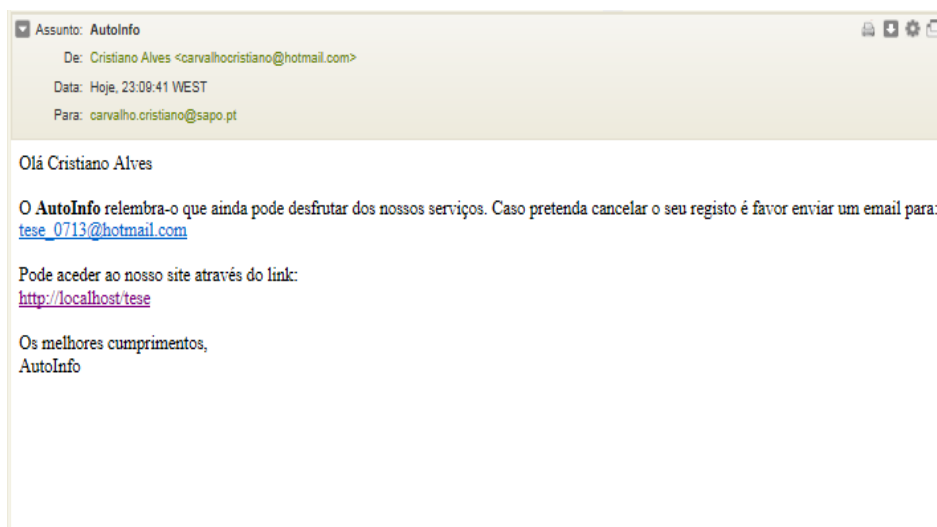



Figura 48 Envio de *email* a relembrar ao utilizador que ainda se encontra inscrito no sítio

4.3.2. EDITAR PERFIL E ALTERAR *PASSWORD*

Após efectuar o registo no sítio o utilizador tem a possibilidade de alterar os seus dados de perfil, corrigindo eventuais dados preenchidos de forma incorrecta ou mudanças do estado de algum dos campos do registo. Para isso, deve ser seleccionada a opção ►***Editar Perfil*** localizada dentro do perfil de utilizador. Surge então a página de edição dos dados de perfil, que apresenta o formato apresentado na Figura 49. Como se pode visualizar ao ser apresentada a página são carregados (a partir da base de dados) os dados de perfil do utilizador e preenchidos automaticamente nos campos respectivos. Estes campos incluem: email, data de nascimento, sexo, cidade, país, carro actual e foto de perfil.

Para alterar os dados o utilizador apenas tem que substituir o(s) valor(es) do(s) campo(s), carregado(s) da base de dados, que deseja modificar, e para concluir o processo e actualizar a base de dados pressionar o botão *Alterar Perfil*. Como se pode observar não é disponibilizado nesta página a alteração da *password*, uma vez que foi criada uma página exclusivamente dedicada a esta operação, que será apresentada de seguida.

Para além de poderem editar os seus dados de perfil os utilizadores podem também alterar a *password* com que efectuaram o registo. Para tal, estes devem seleccionar a opção ►***Alterar Password***, localizada dentro do perfil de utilizador. Assim, surgirá a página respectiva, representada na Figura 50, em que o utilizador terá de inserir a *password* actual e a nova *password* pretendida.

Figura 49 Página para editar os dados de perfil do utilizador

Figura 50 Página para alteração da *password*

Para que seja aceite a nova *password*, esta terá de apresentar um nível de segurança considerado pelo menos como “médio” (ver secção 4.2). Caso cumpra este requisito é enviado um *email* de confirmação para o utilizador, no qual poderá consultar futuramente os novos dados de acesso ao sítio (Figura 51).

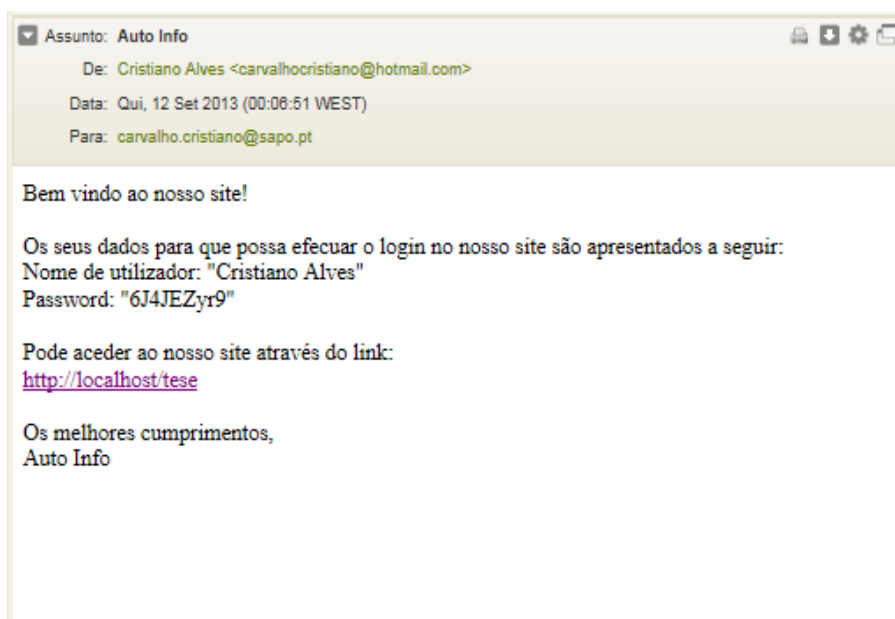



Figura 51 Envio de *email* para confirmação de alteração da *password*

4.3.3. FÓRUM E QUESTIONÁRIO *ONLINE*

Como referido atrás o acesso ao fórum do *site* pode ser efectuado a partir do separador do perfil de utilizador, nomeadamente através da opção ►***Fórum***. Após alguma pesquisa foi optado pelo desenvolvimento do fórum através do serviço *Forumeiros.com* [17], em grande parte devido ao facto de ser gratuito (a hospedagem do fórum também é gratuita) e bastante simples, sendo apenas necessário preencher um formulário com informações básicas e essenciais para a criação do mesmo, conforme exposto na Figura 52.

Como demonstra a Figura 52 ao ser utilizado o serviço *Forumeiros.com* o processo de desenvolvimento de um fórum é bastante simples sendo apenas necessário introduzir no formulário disponibilizado o título, descrição e endereço para o fórum, e o *email* e respectiva senha para administração do mesmo. Após preenchidos estes campos e alteradas algumas opções, disponibilizadas após preenchimento do formulário, o fórum do *site* apresenta o aspecto representado na Figura 53. O fórum criado pode ser acedido através do *link*: *http://autoinfo.forumeiro.com*.

Figura 52 Processo de criação do fórum

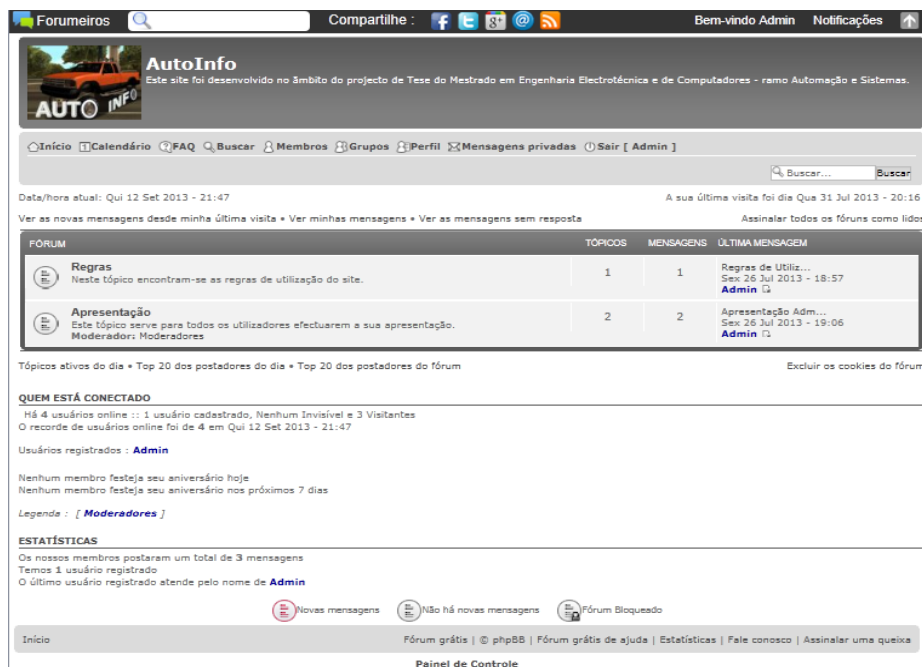



Figura 53 Aspecto final do fórum AutoInfo

O questionário *online* é acedido a partir da opção ►***Questionário Online*** localizada dentro do perfil de utilizador. Na Figura 54 apresenta-se a página relativa a este questionário. Este pretende ser uma área lúdica onde os utilizadores podem testar os seus conhecimentos acerca do "universo automóvel", permitindo ainda ao administrador obter um *feedback* do nível de conhecimento do público-alvo do sítio.

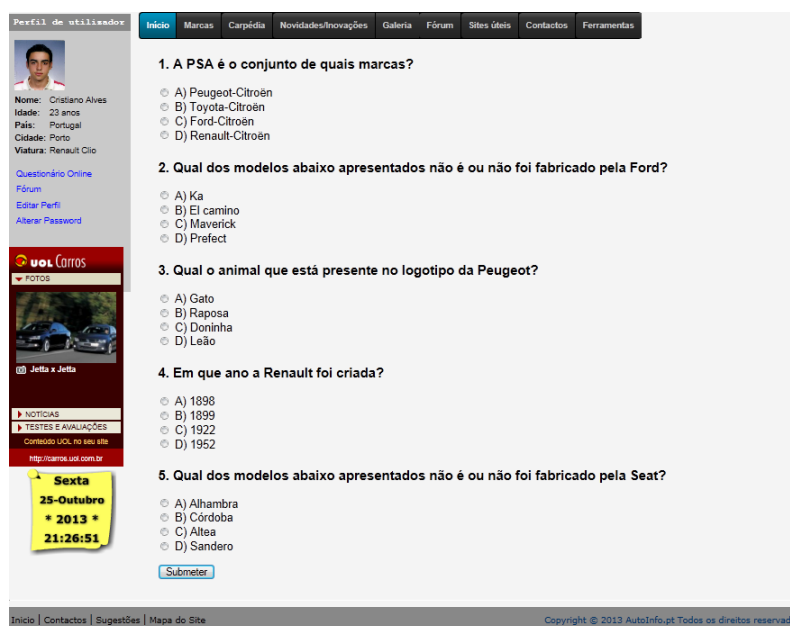



Figura 54 Perfil de Utilizador: questionário *online*

4.4. BARRA DE MENU CENTRAL

A barra de menu central, apresentada na Figura 55, permite que os utilizadores acedam a todas as páginas que fazem parte do *site*. Esta barra é composta por 9 separadores, são eles: *Início*, *Marcas*, *Carpédia*, *Novidades/Inovações*, *Galeria*, *Fórum*, *Sites úteis*, *Contactos* e *Ferramentas*.

O separador apresentado por omissão, isto é, quando se acede ao sítio, corresponde ao separador *Início*. Este por sua vez está associado à página principal do mesmo. A seguir serão apresentados os restantes separadores, com excepção do separador *Fórum* que já foi referenciado na subsecção 4.3.3.

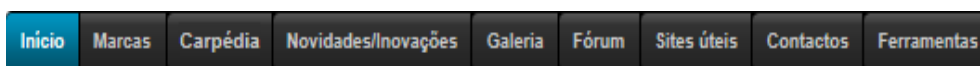



Figura 55 Barra de menu central

4.4.1. SEPARADORES "MARCAS" E "SITES ÚTEIS"

O separador *Marcas* é constituído por 5 opções, que correspondem às marcas de automóveis abordadas neste projecto, isto é, com informação técnica disponibilizada aos utilizadores do sítio. Estas incluem: *citroën*, *ford*, *peugeot*, *renault* e *seat*. Cada uma destas opções encontra-se dividida em quatro itens: *descrição histórica*, *modelos*, *concept cars* e *sites úteis*, como pode ser visualizado na Figura 56. Uma vez que estes itens são comuns a todas as opções incluídas, variando apenas a informação nelas contidas, serão apresentados de seguida apenas os itens referentes a uma destas opções (marcas). Para isto, foi tomado como exemplo a primeira destas marcas, ou seja, a *citroën*.

O item *descrição histórica*, tal como o próprio nome sugere, inclui um enquadramento histórico da marca seleccionada, com apresentação de algumas imagens sugestivas. Este apresenta relevância para os utilizadores que pretendam adquirir algum conhecimento sobre a evolução da marca, desde a sua origem até à actualidade. Na Figura 57 é exposto um exemplo de um destes itens, afecto à marca *citroën*.

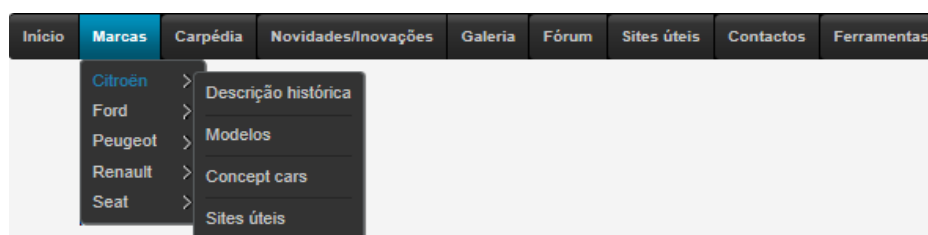



Figura 56 Barra de menu central – separador *Marcas*

Figura 57 Separador *Marcas – descrição histórica*

O item *modelos* é constituído por um formulário em que os utilizadores podem consultar informação técnica acerca de diversos modelos de automóveis. Esta encontra-se dividida em cinco categorias: *geral*, *motor*, *performance*, *dimensões* e *chassi*. Por sua vez, cada categoria, é composta por um conjunto de campos relacionados com esta. Por exemplo, a categoria *geral* é composto pelos campos: *modelo*, *versão*, *tipo de combustível*, *número de portas* e *início de produção*, como demonstra a Figura 58.

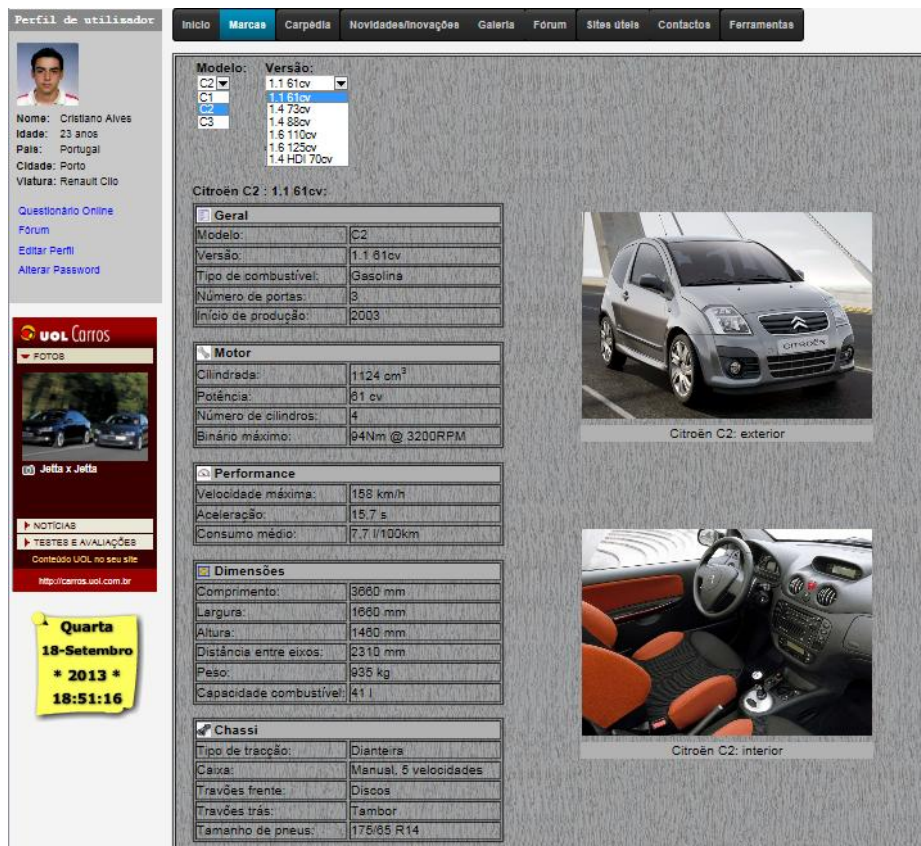



Figura 58 Separador *Marcas – modelos*

O item *concept cars* engloba exemplos de veículos ainda em fase experimental em que são aplicados novos conceitos, sejam eles estéticos, técnicos ou tecnológicos, os denominados carro conceitual ou carro conceito (da expressão inglesa *concept car*). Futuramente, alguns destes veículos poderão dar origem a um veículo de produção em série ou então servir apenas de "exercício de estilo" para os fabricantes demonstrarem as tendências que se supõe virem a ser determinantes para o futuro dos automóveis. Desta forma, ao consultarem esta página os utilizadores podem ficar a par de algumas destas novidades/tendências e da visão de mercado de cada um destes fabricantes. Esta página pode ser consultada na Figura 59.

Conforme pode ser observado na figura acima mencionada, na página relativa ao item *concept cars* é apresentada uma pequena descrição e correspondente imagem alusivas a cada um dos modelos citados. Em cada um destes modelos é ainda incluído um *link* para acesso directo à sua página oficial, intitulado por "*Clique para mais informações*", onde pode ser encontrado uma descrição mais pormenorizada sobre o mesmo.

O item *sites úteis* contém uma lista com os principais *sites* relacionados com a marca seleccionada neste separador. A cada um destes *sites* foi associado um *link* para que os utilizadores possam, caso desejem, aceder a estes, conforme exposto na Figura 60. Nesta lista foram incluídos apenas os *sites* oficiais da marca pesquisada, para não a tornar demasiado extensa e de difícil consulta, e por estes apresentarem, regra geral, informação mais fiável (sem informação adulterada).

Figura 59 Separador *Marcas – concept cars*

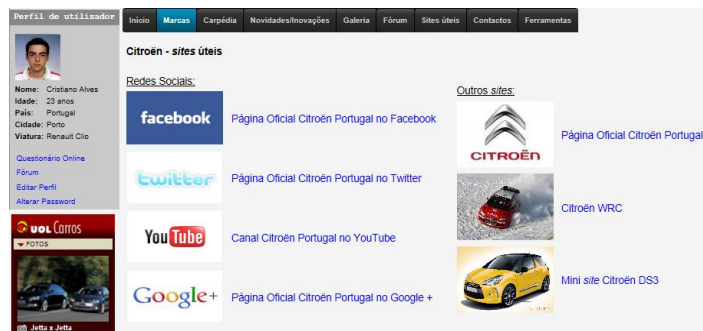



Figura 60 Separador *Marcas – sites úteis*

O separador *Sites úteis* apresenta o mesmo conteúdo do item *sites úteis* (do separador *Marcas*), como demonstra a Figura 61. Contudo este reúne numa única página todas as listas incluídas no sítio (de todas as marcas), divididas por abas (*tags*) que permitem alternar entre elas. Adicionalmente incluiu-se uma aba com *sites* mais gerais com conteúdo de interesse para os "mais entusiastas do mundo automóvel", conforme exposto na Figura 62. À semelhança de algumas das funcionalidades do sítio já apresentadas anteriormente, estas *tags* foram também implementadas em jQuery [25].

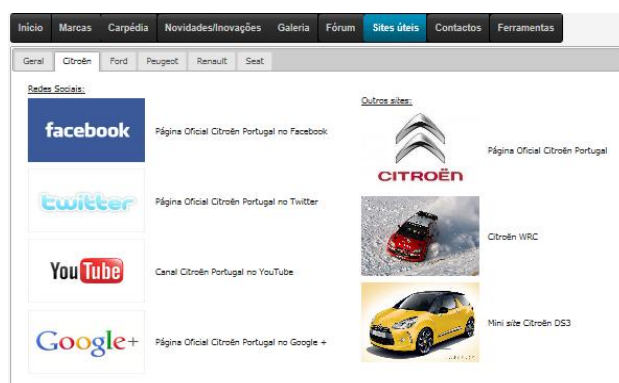



Figura 61 Separador *Sites úteis* – aba *Citroën*

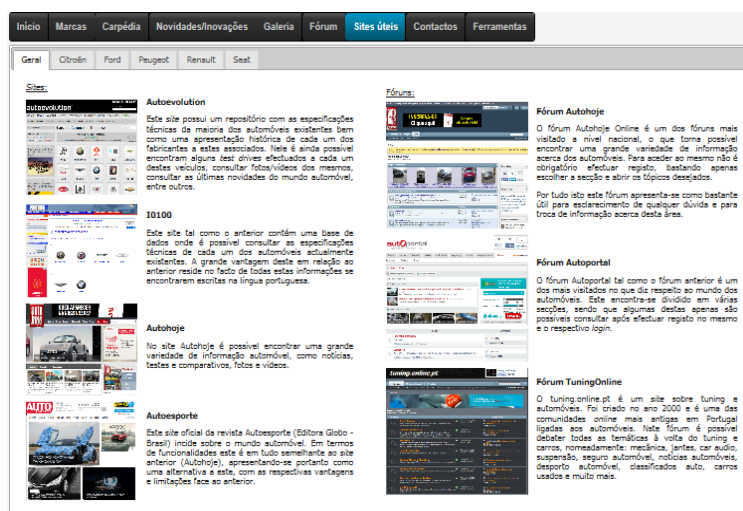



Figura 62 Separador *Sites úteis* – aba *Geral*

4.4.2. SEPARADORES "CARPÉDIA" E "NOVIDADES/INOVAÇÕES"

O separador *Carpédia* foi implementado em *jQuery* e consiste num *accordion*, também conhecido como "menu sanfona" [23], em que é apresentado uma descrição acerca dos diferentes componentes que constituem um automóvel, auxiliada por imagens ilustrativas (Figura 63). Este pretende ser uma fonte de consulta de informação acerca do funcionamento dos automóveis para os utilizadores com maior interesse por esta área.

A *jQuery* é uma biblioteca *Javascript* desenvolvida com o intuito de simplificar os *scripts cliente side* (do lado do cliente) que interagem com o utilizador, sem a necessidade de passar pelo servidor [7]. Esta linguagem é a biblioteca *Javascript* mais utilizada no desenvolvimento de *websites*, sendo actualmente utilizada por cerca de 55% dos dez mil *sites* com maior número de visitas [44].

A utilização do *accordion* serve para estruturar conteúdo de uma página em tópicos, exibindo painéis de conteúdo recolhível para apresentação de informação numa quantidade limitada de espaço. Este é bastante simples de utilizar e pode ser executado com um número reduzido de linhas de código, onde são definidos cabeçalhos que permitem, através do seu clique, expandir/recolher o conteúdo associado a cada um deles, como demonstra a Figura 63 e a Figura 64. O algoritmo desenvolvido para este separador encontra-se exposto no Anexo D.

Figura 63 Separador *Glossário:* cabeçalho "*Motor*"

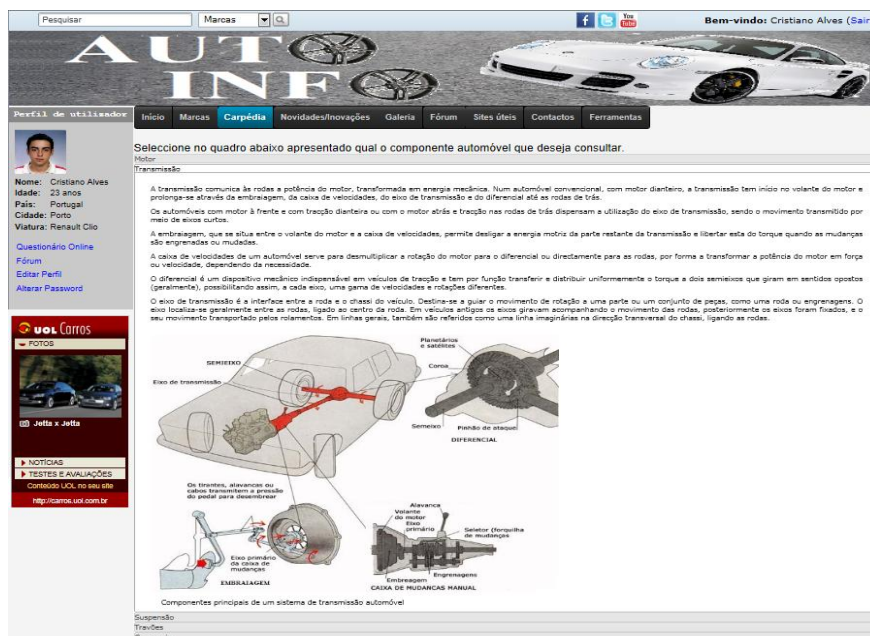



Figura 64 Separador *Glossário:* cabeçalho "*Transmissão*"

O separador *Novidades/Inovações*, à semelhança do separador anterior, foi implementado em *jQuery*, nomeadamente, através de uma janela deslizante (*slider*) [24]. Esta permite apresentar conteúdo destacado no *site* usando animações, nomeadamente notícias com algumas das novidades do mundo automóvel.

Este inclui um título, uma pequena descrição da notícia e uma imagem associada à mesma e um botão para acesso ao *link* da notícia apresentada ("*Ler mais*"), onde esta pode ser lida na sua totalidade, conforme apresentado na Figura 65. É ainda incluído duas classes que permitem alternar entre as notícias apresentadas, avançando ou recuando entre estas, como demonstram a Figura 65 e a Figura 66. O algoritmo desenvolvido para a página associada a este separador, consiste essencialmente no desenvolvimento do *slider*. Este encontra-se representado no Anexo E.

Figura 65 Separador *Novidades/Inovações* – 1ª notícia

Figura 66 Separador *Novidades/Inovações* – 3ª notícia

4.4.3. SEPARADOR "GALERIA"

O separador *Galeria*, como demonstra a Figura 67, encontra-se dividido em 2 sub-separadores: *Fotos* e *Vídeos*.

O sub-separador Fotos, como o próprio nome indica, inclui fotos relativas às diferentes marcas de automóveis utilizadas para o desenvolvimento do sítio. Para exibição destas fotos foi utilizada a aplicação *shadowbox*, já apresentada anteriormente. Assim ao ser aberta (ampliada) uma imagem é criada uma galeria de imagens correspondente à página que esta se encontra, podendo o utilizador retroceder ou avançar a imagem, visualizar o seu título (uma ligeira descrição atribuída à imagem) e encerrar a janela quando pretendido. Esta aplicação encontra-se representada na Figura 68.

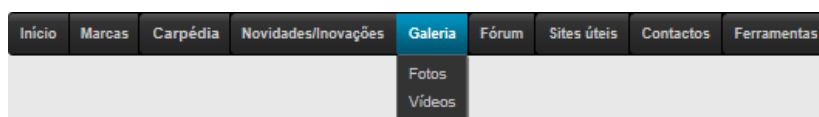



Figura 67 Barra de menu central – separador *Galeria*

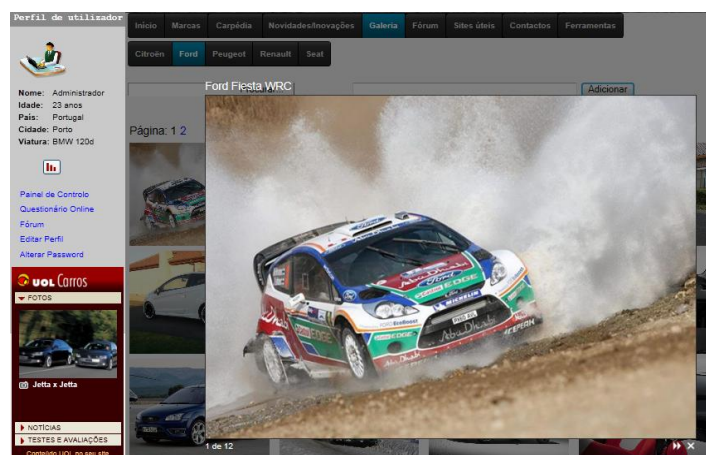



Figura 68 Sub-separador *Fotos* – aplicação da *shadowbox*

Qualquer utilizador pode aceder a este sub-separador, contudo apenas o administrador do sítio pode acrescentar ou remover fotos a este. Para tal, deve efectuar o *upload* das fotos, indicando a qual fabricante de automóveis pertence, de forma a ficar alocada no separador correspondente. Ao inserir a foto o administrador pode também atribuir uma descrição á foto (exemplo: *Ford Fiesta Mk7 Interior*). Na Figura 69 apresenta-se um exemplo da adição de uma nova foto ao sub-separador *Fotos* por parte do administrador. Como pode ser observado as fotos encontram-se distribuídas num formato 3x4 (3 linhas com 4 colunas cada), perfazendo um total de doze fotos por página. Caso uma determinada marca contenha mais de doze fotos, ao ser inserida uma nova, é acrescentada uma nova página onde esta será apresentada (e as onze fotos seguintes que eventualmente possam ser carregadas).

Ao serem introduzidos os dados de uma foto na tabela respectiva da base de dados o algoritmo desenvolvido para este efeito atribui, de forma aleatória, um nome para a imagem, de forma a evitar surgirem imagens sobrepostas caso fossem definidas duas imagens com o mesmo nome. Este nome é armazenado no campo *fotos* da respectiva tabela de dados, como mostra a Figura 70.

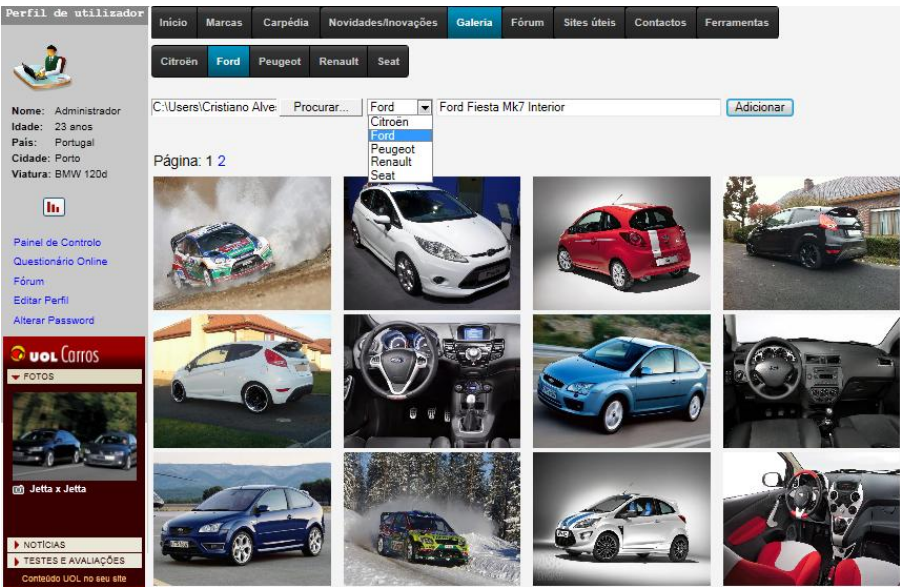



Figura 69   Sub-separador *Fotos* – adicionar uma nova foto

	id_fotos	fotos	pagina	descricao	categoria
	55	95477293f58ff5a170a584f1b4d65746.jpg	2	Ford Fiesta Mk7 Interior	ford

Figura 70   Inserção na base de dados dos dados relativos à foto adicionada

O sub-separador *Vídeos* apresenta um funcionamento semelhante ao sub-separador *Fotos*, e engloba alguns vídeos relativos às marcas de automóveis. Tal como no sub-separador anterior o administrador é responsável pela gestão dos vídeos, no entanto, ao introduzir um novo vídeo à página, este em vez de ser descarregado directamente a partir do seu computador é descarregado a partir de uma ligação ao *Youtube*, através do id atribuído ao vídeo pelo *Youtube*. Na Figura 71 apresenta-se um exemplo da adição de um novo vídeo ao sub-separador *Vídeos* por parte do administrador. À semelhança do anterior também neste sub-separador é utilizada a aplicação *shadowbox*, neste caso para reprodução dos vídeos (Figura 72).

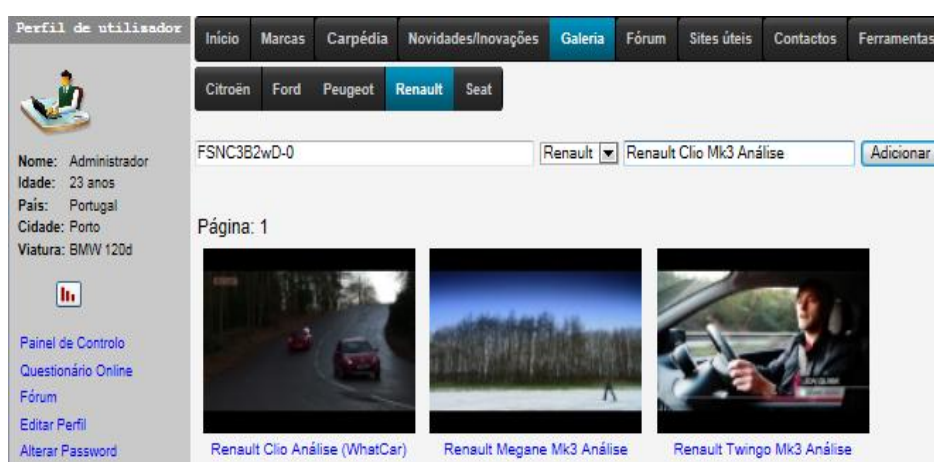



Figura 71 Sub-separador *Vídeos* – adicionar um novo vídeo

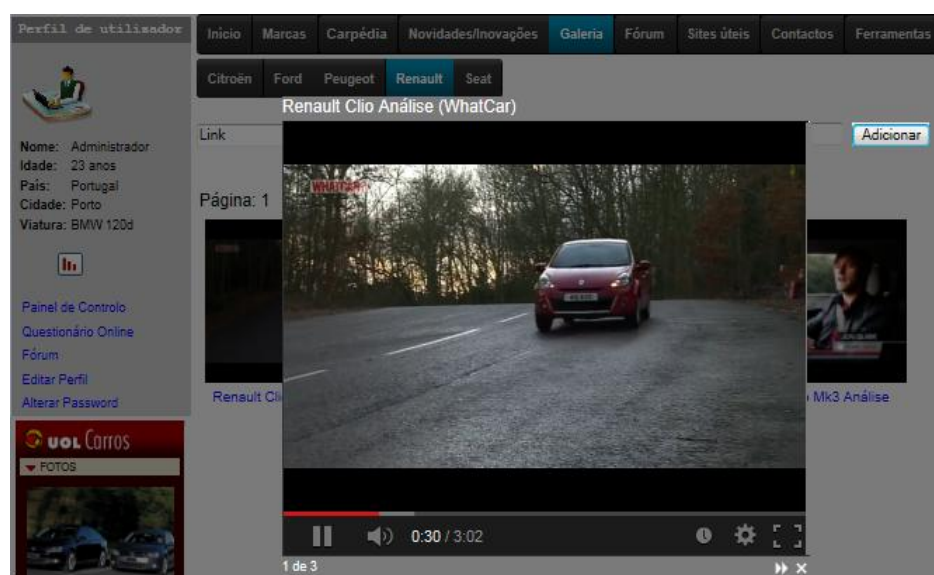



Figura 72 Sub-separador *Vídeos* – aplicação da *shadowbox*

4.4.4. Separador "Contactos"

Se um utilizador necessitar de entrar em contacto com a administração do *site* a fim de esclarecer eventuais dúvidas, efectuar alguma questão acerca dos serviços oferecidos, apresentar sugestões e/ou reclamações, pode fazê-lo através do separador *Contactos*.

Ao aceder a este separador é possível encontrar um formulário, igual ao apresentado na Figura 73, onde é necessário preencher os dados do utilizador (nome e *email*) e os dados da mensagem a ser enviada (assunto e mensagem).

O formulário acima apresentado permite que os utilizadores enviem as suas dúvidas, sugestões e/ou reclamações para o administrador do sítio. Ao preencher este formulário e após submeter o mesmo, através do botão "*Enviar*", é enviado um *email* para o administrador, com a informação preenchida pelo utilizador no formulário (nome, email, assunto e mensagem). Na Figura 74 apresenta-se um exemplo de um *e-mail* recebido pelo administrador com as sugestões e/ou reclamações de um utilizador específico.

O algoritmo responsável por este separador consiste em dois ficheiros: *contactos.php* e *enviar.php*. O primeiro corresponde ao formulário "propriamente dito" e o segundo baseia-se na recolha dos dados enviados pelo formulário e no correspondente envio dos mesmos para o *email* do administrador. Estes ficheiros podem ser consultados no Anexo F. Para o envio destes dados foi utilizada a função *mail*, também apresentada e explicada de forma sucinta no referido anexo, contudo, caso seja pretendido um conhecimento mais aprofundado acerca desta função recomenda-se a leitura do manual da mesma [32].

Figura 73 Separador *Contactos* – formulário de contacto

Assunto: **AutoInfo - Pedido de contacto**
De: Cristiano Alves <carvalhocristiano@hotmail.com>
Data: Hoje, 23:12:49 WEST
Para: carvalho.cristiano@sapo.pt

Uma mensagem vinda do site AutoInfo! Um visitante enviou esta mensagem pelo site:

Nome: Adão Cunha
Email: adaocunha@hotmail.com
Assunto: Sugestão
Mensagem: Olá. Gostava de deixar uma sugestão para o site. Poderiam adicionar mais marcas para consulta. Cumprimentos. Adão Cunha

Figura 74 Envio de *e-mail* com sugestões e/ou reclamações para o administrador do sítio

4.4.5. SEPARADOR "FERRAMENTAS"

Na barra de menu central pode ser encontrado um separador, com um formulário que permite alternar entre diversas funcionalidades, nomeadamente: conversão quilowatt-cavalo vapor (*kw-cv*) e vice-versa, conversão milha por hora-quilómetro por hora (*mph-km/h*) e vice-versa, e cálculo do consumo de combustível, da potência do motor ou do custo monetário de uma viagem, como se apresenta na Figura 75.

A ferramenta *conversão kw-cv;cv-kw* permite aos utilizadores converter um valor em quilowatts para o correspondente valor em cavalo-vapor e vice-versa, conforme apresentado na Figura 75. Esta última unidade não é reconhecida no Sistema Internacional de Unidades (SI), contudo é frequentemente utilizada pelos condutores (e pelos próprios fabricantes de automóveis) para se referirem à potência dos seus veículos.

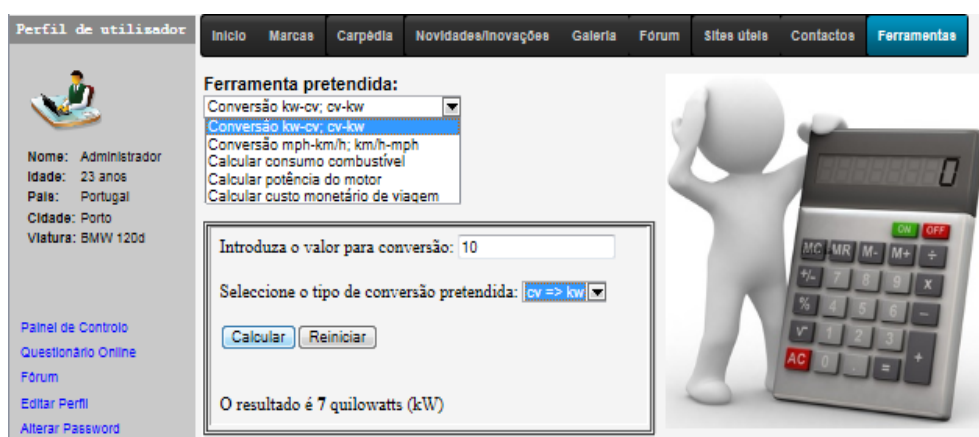



Figura 75 Separador *Ferramentas* – selecção da ferramenta pretendida

Para a ferramenta *conversão kw-cv;cv-kw* foi desenvolvido um algoritmo onde se inclui um formulário com duas opções "*conv1*" e "*conv2*", que correspondem à conversão kw-cv e cv-kw, respectivamente. Após definida a conversão procede-se ao cálculo do valor resultante da conversão. Para o primeiro tipo de conversão ao valor em *kw*, introduzido directamente no sítio, multiplica-se por *1,36*, para obtenção do valor em *cv* respectivo. No segundo tipo de conversão multiplica-se o valor introduzido no sítio por *0,735* para conversão para a unidade *kw* [3]. O algoritmo referido é apresentado de seguida.

```
<form method="post" action="ferramentas_1.php">

Introduza o valor para conversão: <input type="text" name="numero1"/>
Seleccione o tipo de conversão pretendida: <select name="scale">
<option value="conv1">kw => cv</option>
<option value="conv2">cv => kw</option>
</select>
<INPUT TYPE="submit" VALUE="Calcular"/><INPUT TYPE="reset" VALUE="Reiniciar"/>

</form>

<?php
$numero = $_POST['scale'];

(...)

if($numero == "conv1")
   {
      $resultado1 = ($numero1*1.36);
      echo "O resultado é "; .$resultado1; echo " cavalos (cv)";
   }
if($numero == "conv2")
   {
      $resultado2 = ($numero1*0.735);
      echo "O resultado é "; .$resultado2; echo " quilowatts (kW)";
   }
?>
```

A ferramenta *conversão mph-km/h;km/h-mph* apresenta um funcionamento semelhante à ferramenta apresentada anteriormente. Nesta o utilizador tem a possibilidade de introduzir um valor em *mph* e convertê-lo para o respectivo valor em *km/h*. A operação inversa também é possível de ser realizada nesta mesma ferramenta, bastando para tal alterar o valor da *drop-down list* de "*mph=>km/h*" para "*km/h=>mph*" (Figura 76).

Figura 76 Separador *Ferramentas* – ferramenta *conversão mph-km/h;km/h-mph*

A ferramenta *calcular consumo combustível*, assim como o próprio nome indica, permite calcular o consumo de combustível de determinado veículo a partir da quantidade de combustível gasto numa determinada distância percorrida. Esta ferramenta encontra-se representada na Figura 77.

A ferramenta *calcular potência do motor*, representada na Figura 78, possibilita aos utilizadores estimarem a potência de determinado veículo, sendo apenas necessário introduzir alguns dados do mesmo, nomeadamente, o peso e o tempo que este demora a deslocar-se dos 0 km/h até aos 100 km/h. Desta forma os utilizadores podem obter a potência estimada do seu veículo necessitando apenas de introduzir estes dois dados, que podem ser facilmente encontrados no separador *Marcas* já referenciado anteriormente (modelos). Para este cálculo o algoritmo utiliza a equação 1, que a seguir se apresenta. Esta equação foi retirada da Internet, em específico, do *blogue* Freio a Tambor onde é explicado passo a passo como chegar até esta equação [18].

$$Potência\ estimada = \left(\frac{Peso * 769{,}44}{Tempo\ dos\ 0-100km/h}\right) * (1{,}34 * 10^{-3})\ . \quad (1)$$

Figura 77 Separador Ferramentas – ferramenta *calcular consumo combustível*

Figura 78 Separador Ferramentas – ferramenta *calcular potência do motor*

A ferramenta *calcular custo monetário de viagem*, apresentada na Figura 79, permite que os utilizadores possam calcular antecipadamente o valor monetário gasto em combustível associado a uma viagem, constitui-se como uma mais-valia para aqueles utilizadores com maiores preocupações sobre "quanto vão gastar para efectuar determinada viagem". Para isso, os utilizadores devem introduzir no formulário apresentado a distância a ser percorrida, o consumo médio de combustível do automóvel e o seu preço respectivo, que o algoritmo criado encarrega-se de calcular o custo aproximado da viagem. Note-se que caso o utilizador não tenha acesso ao valor actual do preço do combustível para a sua viatura, pode consultar essa informação neste mesmo *site*, uma vez que foi incluído na sua página principal um *script* com esta informação (subsecção 4.1.3).

Figura 79 Separador Ferramentas – ferramenta *calcular custo monetário de viagem*

5. CONCLUSÕES

Com a realização deste trabalho foi possível adquirir novos conhecimentos, e consolidar os já existentes, relativos ao desenvolvimento de aplicações *Web*, nomeadamente, à criação de *websites*. Estes conhecimentos distribuem-se pelas temáticas de sistemas de gestão de base de dados, diagramas UML e linguagens de programação para sistemas computacionais.

Ao elaborar a aplicação *Web* foi possível verificar a importância da execução de uma análise prévia dos requisitos, antes de ser executada a implementação e o *design* das interfaces. Esta análise permitiu verificar quais as necessidades e objectivos que se pretendiam atingir e identificar os principais intervenientes do sistema, isto é, os diferentes tipos de utilizadores (administrador, visitante, utilizador registado), através do diagrama de *use cases*. Os restantes diagramas UML elaborados, diagramas de classes e de entidade-relação, proporcionaram uma melhor compreensão do modelo relacional da base de dados, durante a etapa de desenho e estruturação da mesma.

Aquando do desenvolvimento das várias páginas *Web* que fazem parte do sítio depressa se constatou que a linguagem HTML não era suficiente para atender a todos os requisitos do sistema, tendo sido utilizado código em PHP para execução de *scripts* que permitem interagir com as diferentes funcionalidades do *website*, nomeadamente para tratamento da informação e acesso à base de dados criada. Para a definição do *design* das páginas e inclusão de efeitos visuais foi ainda utilizada *Javascript*.

Após a conclusão deste trabalho, ficou evidente que com todas as ferramentas e *softwares* existentes no mercado para auxiliar o processo de planeamento, desenvolvimento e disponibilização na Internet de um *site*, este é uma tarefa cada vez mais simplificada. O que mais contribuiu para simplificar a implementação do *site*, foi o *software Macromedia Dreamweaver*, pela sua rápida aprendizagem e facilidade de utilização e construção de páginas. Também a utilização do *phpMyAdmin* se mostrou uma mais-valia para o projecto, visto providenciar um conjunto diversificado de serviços de acesso à base de dados, tais como: definição, manipulação e consulta de dados.

Apesar de se entender que este projecto foi bem sucedido, não se pode deixar de referir que nem tudo o que estava projectado foi implementado, ficando apenas a faltar disponibilizar o *website* desenvolvido através da Internet (colocá-lo *online*), não por incapacidade de implementação mas por uma questão de tempo e custo monetário. Desta forma, o custo de implementação do *site* ficou-se apenas pela aquisição dos *softwares Macromedia Dreamweaver* e *phpMyAdmin*. Este configura-se como um ponto a ser adicionado como trabalho futuro. Uma maneira de sustentar os custos de alojamento e manutenção deste potencial melhoramento seria adicionar patrocínios e publicidade ao *site*, preservando sempre o seu bom funcionamento sem incomodar os utilizadores.

Para além desta sugestão, como proposta de melhorias futuras ao sistema destacam-se:

- disponibilização do *site* em diferentes linguagens (multilíngue);
- melhoria da interface gráfica, valorizando mais o trabalho artístico e criativo;
- compatibilização com todos os tipos de navegadores (*browsers*);
- inclusão de novos módulos ou novas tarefas suportadas pelo sistema, conforme se entenda necessário.

Referências Documentais

[1] Adobe Systems – *Download and Installation Help*. Califórnia, Estados Unidos da América, Agosto de 2013, http://helpx.adobe.com/download-install.html

[2] ALVES, Cristiano; ROCHA, Daniel; SILVA, José; SILVA, Marco – GESTÃO DESPORTIVA. Trabalho realizado no âmbito da cadeira de Modelação e Simulação Industrial do Mestrado em Engenharia Electrotécnica e de Computadores, do Instituto Superior de Engenharia do Instituto Politécnico do Porto, em 2012.

[3] Aquidaban – Tabela Geral de Conversão de Unidades. Aquidaban Shipping Trading Logistic & Cargo, São Paulo, Brasil, 2006, http://www.aquidaban.com.br/informacoes

[4] Autoevolution – *Automotive news & test drives*. Bucareste, Roménia, Agosto de 2013, http://www.autoevolution.com

[5] ACP – Automóvel Clube de Portugal, Portugal, Agosto de 2013, http://www.acp.pt

[6] BAPTISTELLA, Francisca; BARCELLINI, Gian Franco – Desenvolvimento de Websites. Centro de Computação da Unicamp, São Paulo, Brasil, 2000.

[7] BENEDETTI, Ryan; CRANLEY, Ronan – *Head First jQuery*. O'Reilly Media, 2011. ISBN: 978-1-4493-9321-2.

[8] BERNERS-LEE, T.; et al. – *The World Wide Web*. Communication of the ACM: Association for Computing Machinery, Nova Iorque, v.37, n.8, p.76-82, 1994.

[9] BOOCH, Grady; JACOBSON, Ivar; RUMBAUGH, James – UML: Guia do Usuário, Tradução da 2ª edição. Editora Campus, Rio de Janeiro, 2005. ISBN 8535217843

[10] BOS, Bert – *Web Style Sheets: Home Page*. W3C Style Activity Lead, Agosto de 2013, http://www.w3.org/Style/

[11] CARMONA, Tadeu – *Curso prático de criação de sites*. Digerati Books, Universo dos Livros Editora Ltda, São Paulo, Brasil, 2006. ISBN: 85-7702-063-0

[12] CONALLEN, Jim – *Building Web Applications with UML*. Addison Wesley, Boston, Estados Unidos da América, 2000. ISBN: 0-201-61577-0.

[13] CSS Globe – *Easy Slider 17 Numeric Navigation jQuery Slider*. Agosto de 2009, http://cssglobe.com/easy-slider-17-numeric-navigation-jquery-slider/

[14] CSS Menu Maker – *jQuery Menus: About our jQuery Menus*, Agosto de 2013, http://cssmenumaker.com/jquery-menus

[15] DANYANG, Cao; BINGRU, Yang – *Design and implementation forMD5-based data integrity checking system*. IEEE 2nd International Conference on Information Management and Engineering, páginas 608-611, Abril de 2010.

[16] DZENDZIK, Isolete – *Processo de desenvolvimento de web sites com recursos da UML*. Dissertação de Mestrado do Curso de Pós-graduação em Computação Aplicada, INPE: Instituto Nacional de Pesquisas Espaciais, São Paulo, Brasil, 2005.

[17] Forumeiros – Descubra que Criar um forum. Forumeiros.com, Agosto de 2013, http://www.forumeiros.com/criar-um-forum

[18] Freio a Tambor – Como Calcular a Potencia do seu Veiculo. Blogue Freio a Tambor, Fevereiro de 2009, http://freioatambor.blogspot.pt/2009/02/como-calcular-potencia-do-seu-veiculo.html

[19] Google – *Google Web Elements*. Google News, Agosto de 2013, http://www.google.com/uds/modules/elements/newsshow/iframe.html?rsz=large&format=300x250&hl=pt-PT&q=Desporto Autom%C3%B3vel

[20] GUEDES, Gilleanes – *UML 2: Uma Abordagem Prática – 2ª edição*. Novatec Editora, São Paulo, Brasil, 2011. 978-85-7522-281-2

[21] ISAAK, Jim - *Web site engineering best practice standards (IEEE 2001)*. Proceedings of the 4th International Workshop on Web Site Evolution, Montreal, Canadá, Outubro de 2002. ISBN: 0-7695-1804-4

[22] JARVINEN, K.; TOMMISKA, M.; SKYTTA, J. – *Hardware Implementation Analysis of the MD5 Hash Algorithm*. Proceedings of the 38th Annual Hawaii International Conference on System Sciences, vol.9, Janeiro de 2005.

[23] jQuery User Interface – *Accordion*. jQuery UI Team, Agosto de 2013, http://jqueryui.com/accordion

[24] jQuery User Interface – *Slider*. jQuery UI Team, Agosto de 2013, http://jqueryui.com/slider

[25] jQuery User Interface – *Tabs*. jQuery UI Team, Agosto de 2013, http://jqueryui.com/tabs

[26] Mais Gasolina – Utilidades: *Widgets*. Portugal, Agosto de 2013, http://www.maisgasolina.com/widgets/

[27] MICROSOFT – *The Official Microsoft ASP.NET Site*. Neudesic, LLC. (Microsoft), Califórnia, 2013, http://www.asp.net/

[28] OLIVEIRA, João N.; SANTOS, Leonel D. dos; Amaral, Luís A. do – *Guia de Boas Práticas na Construção de Web Sites da Administração Directa e Indirecta do Estado*. Laboratório de Estudo e Desenvolvimento da Sociedade da Informação, Universidade do Minho, Portugal, 2003. ISBN: 972-98921-0-5.

[29] ORACLE – *JavaServer Pages Technology*. Oracle Corporation, Califórnia, 2013, http://www.oracle.com/technetwork/java/javaee/jsp/index.html

[30] Ultimate Specs – *Car Specifications*. Agosto de 2013, http://www.ultimatespecs.com/pt

[31] PHP – *PHP: Downloads*. The PHP Group, Outubro de 2013, http://www.php.net/downloads.php

[32] PHP – *Php Manual: mail*. The PHP Group, Agosto de 2013, http://www.php.net/manual/pt_BR/book.mail.php

[33] PhpMyAdmin – *phpMyAdmin Download*. The phpMyAdmin Project, Agosto de 2013, http://www.phpmyadmin.net/home_page/downloads.php

[34] PIMENTA, Marcelo; WINCKLER, Marco – *Avaliação de Usabilidade de Sites Web*. Instituto de Informática, Universidade Federal do Rio Grande do Sul, Brasil, 2002.

[35] PortalF1 – Últimas Notícias da F1, Reino Unido, Agosto de 2013, http://www.portalf1.com/pt

[36] SANTOS, Roneclei; JORGE, Eduardo – *Java Server Pages - JSP*. Javafree.org, 2008, http://javafree.uol.com.br/files_user/files/A/74/F8/Tutorial_JSP1.pdf

[37] SANTOS, Verônica S. dos – *Arquitetura e Metodologia na Criação de WebSite*. Trabalho de conclusão de Curso para obtenção do grau de Licenciado em Tecnologia em Desenvolvimento de Software, Centro Universitário Amparense, Brasil, 2007.

[38] SCAPIN, Dominique et al. – *Transferring Knowledge of User Interfaces Guidelines to the Web*. Annual Meeting of the Special Interest Group, Springer London, páginas 293-304, 2001. ISBN: 978-1-85233-355-3.

[39] Shadowbox – *Shadowbox Download page*. Agosto de 2013, http://www.shadowbox-js.com/download.html

[40] SIQUEIRA, Bruno Rodrigues – *Apostila de PHP*. Outubro de 2013, http://www.etelg.com.br/paginaete/downloads/informatica/apostila_php.pdf

[41] Tuning Online – O portal do *tuning* e automóvel, Portugal, Agosto de 2013, http://www.tuning.online.pt

[42] VIEIRA, João – *Programação Web com Active Server Pages - 1ª edição*. Edições Centro Atlântico, Portugal, Janeiro de 2000. ISBN: 972-8426-21-6

[43] ZAFIRIS, A. P.; GEORGANTIS, P. N. – *A Practitioner's Approach to Evolving and Remodeling Large-Scale WWW Sites*. IEEE: 34th Hawai International Conference on System Sciences, Maui, Hawai, 3-6 de Janeiro, 2001.

[44] W^{3}Techs – *Usage of JavaScript libraries for websites*. Q-Success Web-based Services, Agosto de 2013, http://w3techs.com/technologies/overview/javascript_library/all0

Anexo A. Algoritmo: “*coluna da direita*”

Neste anexo é apresentado o extracto de código referente à “*coluna da direita”*, isto é, à apresentação das classificações, individual e por equipas, do campeonato de Fórmula 1, época 2012/2013.

```
<?php

$tipo_classificacao = $_GET["tipo_classificacao"];
if($tipo_classificacao == "")
 $tipo_classificacao = "individual";

/************ Criar tabela ***************/

$sql = "SELECT posicao, piloto, equipa, pontos FROM
formula1_classificacao";
$limite = mysql_query("$sql"); //verifica o limite da tabela

$posicao = $sql["posicao"];
$piloto  = $sql["piloto"];
$equipa  = $sql["equipa"];
$pontos  = $sql["pontos"];

//loop para exibir os dados da tabela
while ($sql = mysql_fetch_array($limite))
 {
 if($tipo_classificacao == "individual")
       {
       echo "$posicao";
       echo "$piloto";
       echo "$pontos";
       }

 if($tipo_classificacao == "equipas")
       {
       echo "$posicao";
       echo "$equipa";
       echo" $pontos";
       }
 }

?>
```

Conforme se pode visualizar no extracto de código acima apresentado, em primeiro lugar é recolhido, através da variável *$_GET*, o valor do tipo de classificação do formulário e armazenado em *$tipo_classificacao*. A seguir são seleccionados os campos da tabela de dados que se pretende utilizar, nomeadamente, “*posicao”*, “*piloto”*, “*equipa”* e “*pontos"*, e armazenados individualmente nas variáveis $*posicao*, $*piloto*, $*equipa* e $*pontos*,

respectivamente. Esta selecção é realizada através da instrução SQL: *$sql = "SELECT posicao, piloto, equipa, pontos FROM formula1_classificacao"*. Por fim, o programa avança para um ciclo (*loop*) onde os dados são impressos para o ecrã até que seja atingido o limite da tabela, isto é, até serem exibidos todos os dados (*$limite*). Se tiver sido seleccionado no formulário a classificação individual (*$tipo_classificacao == "individual"*) são impressos os valores contidos nas variáveis $*posicao*, $*piloto* e $*pontos*. Se for pretendida a consulta da classificação por equipas (*$tipo_classificacao == "equipas"*) são impressas as variáveis $*posicao*, $*equipa* e $*pontos*.

Anexo B. Algoritmo: análise de *passwords* e verificação do nível de segurança

Neste anexo apresenta-se a função responsável pela análise da *password* introduzida por um utilizador na página de registo como sua password de autenticação no *site*. Esta função, *passwordStrength(password)*, permite ainda verificar o nível de segurança associada à *password* introduzida.

```
function passwordStrength(password)
{
      var vector = new Array();
      vector[0] = "Muito Fraca";
      vector[1] = "Fraca";
      vector[2] = "Melhor";
      vector[3] = "Média";
      vector[4] = "Forte";
      vector[5] = "Muito Forte";

      var nivel  = 0;

  //Se password > 4 caracteres, incrementa vector
      if (password.length > 4) nivel++;


  //Se password tiver pelo menos um caracter maiúsculo, incrementa vector
      if ((password.match(/[a-z]/)) && (password.match(/[A-Z]/))) nivel++;


  //Se password tiver pelo menos um número, incrementa vector
      if (password.match(/\d+/)) nivel++;


  //Se password > 6 caracteres, incrementa vector
      if (password.length > 6) nivel++;


  //Se password > 8 caracteres, incrementa vector
      if (password.length > 8) nivel++;


document.getElementById("passwordDescription").innerHTML = vector[nivel];
document.getElementById("passwordStrength").className = "strength" + nivel;
}
```

A função apresentada contém um variável *nivel* inicializada a zero e que vai sendo incrementada à medida que atravessa os testes efectuados (caso falhe um teste não é incrementado o seu valor). Quantos mais condições (“*if*”) forem verdadeiras maior será o

valor armazenado nesta variável (de *0* a *5*). No final do algoritmo o valor desta variável "apontará" para a posição respectiva do vector *vector*.

Para que a *password* seja aceite durante o processo de registo do utilizador o seu nível de segurança terá de ser considerado pelo menos como médio, ou seja, verificar pelo menos 3 destas condições, a que corresponde a posição 3 do vector (*vector[3] = "Média"*).

Anexo C. Algoritmo: gerar aleatoriamente nova *password*

Neste anexo é apresentado o algoritmo responsável pela geração, de forma aleatória, de uma nova *password*. Este é executado sempre que solicitado uma nova *password* de acesso por parte de um dos utilizadores, devido ao esquecimento da mesma.

```
<?php

if($_POST['enviar_nova_password'])
{
 $email = $_POST['email'];

 $sql = "SELECT password, email FROM utilizadores WHERE email='$email'";

 if($email == 'email')
 {
   $CaracteresAceitos =
'abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789';
   $max = strlen($CaracteresAceitos)-1;
   $pass = null;

   for($i=0; $i < 8; $i++) {
       $pass = $CaracteresAceitos{mt_rand(0, $max)};
   }
   $passd = md5($pass);

   $sqlinsert = "UPDATE utilizadores SET password='$passd' WHERE email='$email'";
   mysql_query($sqlinsert) or die("Não foi possível inserir os dados");
   require("../registar/enviar_email.php");   // Ficheiro para enviar email
   ?> <b id="label">A sua password foi enviada com sucesso.</b> <?php
 }

 else
 {
   ?> <b id="erro">O email que inseriu não se encontra registado!</b> <?php
 }
}

?>
```

Quando se pressiona o botão “Enviar nova password” o algoritmo começa por recolher o *email* inserido e armazena-o na variável *$email*, sendo a seguir seleccionados, a partir da tabela que contem os dados dos utilizadores (*utilizadores*), os campos *password* e email que correspondem a esse email. Esta selecção é realizada através da instrução SQL: *$sql = "SELECT password, email FROM utilizadores WHERE email= '$email'"*.

Se for encontrada correspondência entre o *email* inserido no *site* com um dos *emails* registados na tabela de dados é criado um *array* com os caracteres válidos e o código “transita” para o ciclo *for*, onde são executadas 8 iterações (de “*i = 0*” até “*i < 8*”), em que é

gerado aleatoriamente um caracter em cada iteração ao ser executado um *random* ao *array* criado. Este *random* é feito com recurso à função *mt_rand*, que apresenta um funcionamento semelhante à função rand, porém capaz de atribuir números aleatórios num intervalo de tempo quatro vezes inferior. No final destas iterações a variável *$pass* contém a nova *password* do utilizador. Contudo é ainda necessário encriptar a *password* antes de a actualizar na base de dados, sendo portanto utilizada a função md5 (*Message-Digest algorithm 5*).

Por fim, actualiza-se o valor do campo *password* na tabela de dados, de forma a corresponder com a nova password que será enviada para o utilizador *($sqlinsert = "UPDATE utilizadores SET password='$passd' WHERE email='$email'"*). Em caso de falha ao inserir os novos dados na tabela é apresentada uma mensagem de aviso "*Não foi possível inserir os dados*". Caso contrário é solicitado o ficheiro *enviar_email.php*, responsável pelo envio do *email* com a nova *password*.

Anexo D. Algoritmo: separador “*Carpédia*”

Neste anexo apresenta-se o algoritmo associado ao separador “*Carpédia*”, ou seja, à página de *Carpédia*.

```
<html>

<head>
<title>Carpédia</title>

   (...)

   <script>
       $(function() {
         $("#accordion").accordion();
       });
   </script>
</head>


<body>

Seleccione no quadro abaixo apresentado qual o componente automóvel que deseja consultar.

<div id="accordion">
   <h3>Motor</h3>
       <div>
              (...)
              <!-- Informação acerca do componente "Motor" -->
       </div>

   <h3>Transmissão</h3>
       <div>
              (...)
              <!-- Informação acerca do componente "Transmissão" -->
       </div>

   <h3>Suspensão</h3>
       <div>
              (...)
              <!-- Informação acerca do componente "Suspensão" -->
       </div>

   <h3>Travões</h3>
       <div>
              (...)
              <!-- Informação acerca do componente "Travões" -->
       </div>

   <h3>Carroçaria</h3>
       <div>
              (...)
              <!-- Informação acerca do componente "Carroçaria" -->
       </div>
</div>

(...)

</body>

</html>
```

O algoritmo apresentado acima, é constituído por um cabeçalho (identificado pelas *tags* *<head>* e *</head>*), que fornece informações relativas à página e por um corpo principal (identificado pelas *tags* *<body>* e *</body>*) que engloba os elementos que serão exibidos no navegador.

No cabeçalho foi definido, através do elemento *<title>*, um título para a página, que é exibido na barra de títulos do navegador usado. Este inclui ainda um *script* onde é incluída a função *JavaScript* onde se encontra toda a programação necessária para o funcionamento do menu *accordion*.

No corpo principal é apresentado o conteúdo do menu *accordion*. Este foi dividido em cinco cabeçalhos (*<h3>*) e respectivos conteúdos *div*, que correspondem aos diferentes componentes que constituem um automóvel e que neste foram abordados, são eles: motor, transmissão, suspensão, travões e carroçaria. Estes cabeçalhos permitem, através do seu clique, expandir/recolher o conteúdo associado a cada um deles (incluído dentro das *tags* *<div>* e *</div>*).

Anexo E. Algoritmo: separador “*Novidades/Inovações*”

Neste anexo é apresentado o algoritmo desenvolvido para a página associada ao separador *Novidades/Inovações*.

```
<html>
(...)
<body>
<div class="container">
   <div id="da-slider" class="da-slider">
       <div class="da-slide">
          <h2>Volvo apresenta carregador eléctrico...portátil!</h2>
               <p align="justify">Chama-se «Pure Tension Pavillion» (...)</p>
               <a href="link_da_notícia" class="da-link">Ler mais</a>
               <a  href="link_da_imagem"><img  src="images/1.jpg"  alt="image01"
width="340" height="235"/></a></div>
       </div>

       <div class="da-slide">
           <h2>Renault Fluence Z.E. é táxi em Lisboa</h2>
               (...)
               <!-- Texto e imagem-->
       </div>

       (...)
       <div class="da-slide">
           <h2>Portuguesa desenvolve tecnologia «vital» para carros elétricos</h2>
               (...)
               <!-- Texto e imagem-->
       </div>

       <nav class="da-arrows">
           <span class="da-arrows-prev"></span>
           <span class="da-arrows-next"></span>
       </nav>
   </div>
</div>
             (...)
<script type="text/javascript">
   $(function() {
       $('#da-slider').cslider({
             autoplay      : true,
             bgincrement   : 450,
             (...)
       });
   });
</script>
(...)
</body>
</html>
```

O extracto de código acima apresentado consiste essencialmente no desenvolvimento de um *slider*. O *slider* desenvolvido é composto por quatro classes “*da-slide*”, o que

corresponde a uma por cada notícia. Cada classe engloba um título (cabeçalho *<h2>*), uma descrição da notícia (entre as *tags* *<p align="justify">* e *</p>*), um *link* para acesso à notícia (*<a href="link_da_notícia">*) e um outro *link* para acesso à foto (*<a href="link_da_imagem">*). A classe "*da-arrows*" permite alternar entre as notícias apresentadas, avançando (*da-arrows-next*) ou recuando (*da-arrows-prev*) entre as mesmas.

Por fim é incluída a função responsável pelo movimento deslizante do *slider*. Os parâmetros mais relevantes para o desenvolvimento do *slider* foram: *autoplay* e *bgincrement*. Ao ser definido como verdadeiro (*true*) o parâmetro *autoplay* o *slider* irá percorrer automaticamente as classes, quando carregada a página das novidades/inovações, isto é, sem necessidade do utilizador carregar nos botões de avançar ou recuar as notícias. O parâmetro *bgincrement* permite incrementar a posição da base do *slider* (*background*), neste caso definido como *450* o que corresponde ao tamanho total de cada uma das classes, permitindo que a cada movimento seja alterada a classe exibida.

Anexo F. Algoritmo: separador “*Contactos*”

Neste anexo expõe-se o algoritmo criado para o envio de sugestões e/ou reclamações, por parte dos utilizadores do sítio desenvolvido, ao administrador. Este algoritmo encontra-se “dividido” em dois ficheiros: *contactos.php* e *enviar.php*.

O ficheiro *contactos.php* corresponde ao formulário desenvolvido para este efeito. O código (extracto) presente neste é apresentado a seguir:

```
<html>

<head>
    <title> Contactos </title>
    (...)
</head>

<body>
(...)

<form name="form1" method="post" action="enviar.php">

     <p> Nome: <input type="text" name="nome" id="nome”> </p>

     <p> Email: <input type="text" name="email" id="email"> </p>

     <p> Assunto: <input type="text" name="assunto" id="assunto"> </p>

     <p>  Mensagem:  <textarea  name="mensagem"  wrap="VIRTUAL"  id="mensagem" cols="25" rows="10"></textarea> </p>

     <p> <input type="submit" name="Submit" value="Enviar"> </p>

</form>

(...)

</body>

</html>
```

O formulário presente no ficheiro *contactos.php* é identificado pelo marcador *<form name="form1" method="post" action="enviar.php">*. O atributo *action* representa o programa que irá tratar os dados do formulário, neste caso, o ficheiro *enviar.php*. O *method* especifica o método pelo qual os dados serão enviados, tendo sido utilizado o método post, uma vez que neste os dados são enviados à parte, ou seja, separados do cabeçalho da requisição que o navegador envia para o servidor. Desta forma, os dados inseridos no formulário serão passados para a página enviar.php de uma forma “invisível” para o utilizador, visto que as variáveis enviadas não são mostradas na URL (*Uniform Resource Locator*).

Dentro do formulário foram definidos quatro campos: nome, *email*, assunto e mensagem. Os três primeiros correspondem a campos com entrada de dados de apenas uma linha (*<input type="text">*) e o último com entrada de dados de várias linhas (*textarea*). Existe ainda um botão *submit* para proceder ao envio destes dados.

O ficheiro *enviar.php* permite recolher os dados do formulário e enviá-los para o *email* do administrador. O código respectivo a este ficheiro é apresentado a seguir:

```
<html>

(...)

<?php
//declaração das variáveis que recebidas pelo método post
$email_destino = 'carvalhocristiano@hotmail.com';
$nome=$_POST['nome'];
$email=$_POST['email'];
$assunto=$_POST['assunto'];
$mensagem=$_POST['mensagem'];

$corpoDoEmail="
Uma mensagem vinda do site AutoInfo! Um visitante enviou esta mensagem pelo site:
Nome: $nome Email: $email Assunto: $assunto Mensagem: $mensagem";

$cabecalho = "MIME-Version: 1.0" . "\r\n".
             "Content-type: text/html; charset=iso-8859-1" . "\r\n".
             "From: $email" . "\r\n" .
             "Reply-To: $email_destino";

$assunto2 = "AutoInfo - Pedido de contacto";

//enviar os dados usando a função mail
mail($email_destino, $assunto2, $corpoDoEmail, $cabecalho);

//Mostrar no navegador da pessoa que enviou o email uma mensagem de confirmação
echo "A sua mensagem foi enviada com sucesso!";
?>

(...)

</html>
```

Tal como é possível observar no extracto de código apresentado acima, em primeiro lugar são recolhidos, através da variável *$_POST*, os dados inseridos no formulário e armazenados nas variáveis correspondentes: *$nome*, *$email*, *$assunto* e *$mensagem*. Por último, são enviados os dados para o *e-mail*, através da função *mail*. O primeiro parâmetro *$email_destino* refere-se ao receptor do *e-mail*, neste caso o administrador do sítio (*carvalho.cristiano@sapo.pt*). O parâmetro *$assunto2* designa o assunto do e-mail a ser enviado. O parâmetro seguinte *$corpoDoEmail* diz respeito à mensagem em si, ou seja, à mensagem que será enviada para o administrador. Por fim, o parâmetro *$cabecalho* serve para adicionar cabeçalhos extra (como por exemplo: *From*, *Cc*, e *Bcc*).